mathematics

MDPI

Article

Optimizing a Multi-Layer Perceptron Based on an Improved Gray Wolf Algorithm to Identify Plant Diseases

Chunguang Bi [1], Qiaoyun Tian [1,*,†], He Chen [1], Xianqiu Meng [2], Huan Wang [3], Wei Liu [3] and Jianhua Jiang [4]

[1] College of Information Technology, Jilin Agricultural University, Changchun 130118, China; chunguangB@jlau.edu.cn (C.B.); 15041912010@163.com (H.C.)
[2] College of Computer Science and Technology, Jilin University, Changchun 130118, China; wangyimxq458@163.com
[3] College of Foreign Languages, Jilin Agricultural University, Chunchun 130118, China; wanghuan@jlau.edu.cn (H.W.); liuwei6467@jlau.edu.cn (W.L.)
[4] Center for Artificial Intelligence, Jilin University of Finance and Economics, Changchun 130118, China; jjh@jlufe.edu.cn
* Correspondence: t15500183867@163.com
† Qiaoyun Tian has made a major contribution to this paper.

Abstract: Metaheuristic optimization algorithms play a crucial role in optimization problems. However, the traditional identification methods have the following problems: (1) difficulties in nonlinear data processing; (2) high error rates caused by local stagnation; and (3) low classification rates resulting from premature convergence. This paper proposed a variant based on the gray wolf optimization algorithm (GWO) with chaotic disturbance, candidate migration, and attacking mechanisms, naming it the enhanced gray wolf optimizer (EGWO), to solve the problem of premature convergence and local stagnation. The performance of the EGWO was tested on IEEE CEC 2014 benchmark functions, and the results of the EGWO were compared with the performance of three GWO variants, five traditional and popular algorithms, and six recent algorithms. In addition, EGWO optimized the weights and biases of a multi-layer perceptron (MLP) and proposed an EGWO-MLP disease identification model; the model was tested on IEEE CEC 2014 benchmark functions, and EGWO-MLP was verified by UCI dataset including Tic-Tac-Toe, Heart, XOR, and Balloon datasets. The experimental results demonstrate that the proposed EGWO-MLP model can effectively avoid local optimization problems and premature convergence and provide a quasi-optimal solution for the optimization problem.

Keywords: swarm intelligence; GWO; EGWO; multi-layer perceptron; EGWO-MLP disease identification model

MSC: 68T20; 68T07

Citation: Bi, C.; Tian, Q.; Chen, H.; Meng, X.; Wang, H.; Liu, W.; Jiang, J. Optimizing a Multi-Layer Perceptron Based on an Improved Gray Wolf Algorithm to Identify Plant Diseases. *Mathematics* **2023**, *11*, 3312. https://doi.org/10.3390/math11153312

Academic Editor: Gaige Wang

Received: 15 June 2023
Revised: 7 July 2023
Accepted: 12 July 2023
Published: 27 July 2023

1. Introduction

The agrochemical network reported that an informal meeting of EU agriculture ministers was held in Prague. The conference's theme was food security, the role of European agriculture, and food in global sustainable food production. Conflicts between Russia and Ukraine, the lingering effects of the COVID-19 pandemic, and rising climate change are majorly impacting global food security and prices. As the world's population grows, more food must be produced to increase sustainable agricultural production and reduce food waste.

Farmers face crop disease management and prevention issues [1]. Due to the diversity and complexity of soybean diseases, they are more likely to appear in large areas under certain conditions, resulting in escalating soybean yield reductions. The impact of soybean disease increases with scale, but identifying and evaluating the crop's final yield through improved and enhanced disease models remain a challenge [2]. During the growth of

soybean, a variety of leaf diseases often occur, affecting the yield and quality of the soybean, which are also major factors in destroying crop health and causing genetic mutations. Cell mutations or tissue damage can lead to reduced yields and even crop extinction [3]. Agricultural diseases can seriously affect crop growth and threaten food security. Timely spraying of pesticides can control the spread of diseases and is one of the main measures to reduce losses [4]. Soybean acreage and planting methods are constantly changing. Conservation tillage operations such as intercropping, environmental disease of straw, and returning farmland to farmland are increasing, which makes it more difficult to predict and control diseases [5]. Rapid and accurate diagnosis of crop diseases will enable measures to be taken to improve their overall management and the effective prevention and control of the diseases [6]. Soybean leaf diseases are shown in Figure 1.

Figure 1. Soybean diseases.

The gradual emergence of food crop diseases has attracted significant attention from various countries. Therefore, conducting in-depth research on the identification and effective and accurate control of crop diseases is of great significance. Learning-based image processing techniques are often used for crop disease diagnosis and recognition [7]. Traditional image recognition technology based on manually collecting image features will affect the model's overall classification and recognition performance [8]. Agricultural intelligent detection based on the Internet of Things (IoT) and artificial intelligence (AI) is aimed at monitoring and detecting diseases [9].

With the development of group intelligence, artificial intelligence, and intelligent agriculture, scholars have researched crop disease identification, crop image processing

based on computer technology, and crop disease identification technology [10]. Pests and pathogens enter crops and produce externally visible traits. Neural networks use deep convolutional migration to identify foliage diseases of crops [11]. Semantic segmentation uses pictures of dead plant leaves to identify disease [12]. Image processing strategies have recently been used widely and increasingly in agriculture due to their excellent characteristics, including accuracy, speed, and cost sensitivity [13].

Artificial neural networks (ANNs), called neural networks (NNs) or connectivity models, are algorithmic mathematical models with distributed parallel information processing that mimic the behavioral characteristics of animal neural networks. Neural networks have multiple and single layers. Each layer contains several neurons connected by a directed arc with variable weights. The network is trained by iterative learning of known information, which is changed by gradually adjusting the connection weights of neurons to process information and simulate input–output relationships. NNs are commonly used in machine learning [14]. For example, NNs can be used for image identification [15], speech identification [16], and so on, and can be extended to other fields. Computer vision and deep learning advances can predict impending crop disease [17]. Examples of common NNs are backpropagation (BP) [18] convolutional neural networks (CNNs) [19,20]. The most classic neural network is the multi-layer perceptron (MLP) [21]. The advantages of MLP can be summarized as follows: (1) high parallel processing; (2) highly nonlinear global effect; (3) good fault tolerance; (4) an associative memory function; (5) an adaptive solid and self-learning function.

There has been a tendency to use MLP to structure identification models, with remote sensing spectral imaging based on the MLP-CNN classifier [22]. Swarm intelligence optimization algorithms have been applied to optimize the weights and biases of the MLP, such as the PSO-based back propagation learning MLP [23], the application layer attack detection algorithm based on MLP-GA [24], and neural-based electric load forecasting using hybrid feature selection of GA and ACO [25].

The swarm intelligence optimization algorithm is a new computing intelligence technology, shown in Figure 2, which has become the focus of increasingly more scholars. It originates from the simulation of biological evolution processes or behavior in nature. The objective function measures individual adaptability to the environment. The survival of the fittest or individual foraging is compared. It finds the best agents and replaces poorly feasible solutions with reasonable, achievable solutions in the iterative optimization process. Forming a search algorithm with the characteristics of "generation + testing" is a method to solve the optimization problem based on adaptive artificial intelligence technology.

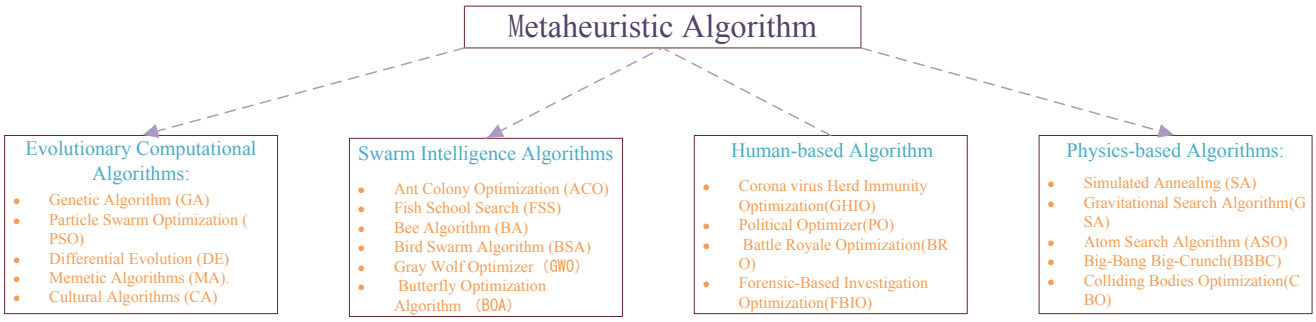


Figure 2. Metaheuristic Optimization Algorithms (MOAs).

The gray wolf optimizer (GWO) [26] is an efficient optimization algorithm that imitates the wolf hierarchy and has a more vital ability to approximate the global optimum. In the GWO, gray wolves are generally divided into four levels and the α wolf, β wolf, δ wolf, and ω wolf represent the gray wolves of the first, second, third, and fourth levels, respectively. It has been successfully applied to solve feature subset selection [27], optimization of MLP weights and biases [28], etc. Scholars have been studying it and have proposed several variants, such as MGWO [29] and binary GWO with the integration of DE and

GWO [30], the hybrid version of GWO and PSO [31]. However, No Free Lunch (NFL) states that no single heuristic algorithm can solve all optimization problems in different scenarios [32], which means newly proposed hybrid optimization algorithms can only solve some problems. Some heuristics can enhance their ability to relieve local optima, but there is still premature convergence [33]. Then, although some of the above variants have been proposed, there is room for improvement. In this regard, the motivations of the present study are as follows.

1.1. Motivations behind the Present Work

Although the GWO algorithm has certain advantages in some aspects, it also has some disadvantages:

1. Slow convergence speed;
2. Affected by the initial parameters;
3. Poor effect on high-latitude problems.

Therefore, the performance of the GWO should be guaranteed to become infinitely close to the global optimal solution and alleviate local stagnation.

- The performance of the GWO should be ensured to become infinitely close to the global optimal solution.
- Local stagnation should be relieved.
- The possible defects of MLP include overfitting, difficulty in determining the optimal structure, long training times, and the ease of falling into a locally optimal solution. The improved GWO-MLP alleviates these problems and increases the optimization capability of the MLP to improve the classification rate and alleviate local stagnation.

1.2. Contribution of This Study

In order to further improve the performance of the GWO algorithm and make up for its shortcomings, the weight and biases of the MLP are modified by adding the enhanced GWO. Introducing MLP into the GWO algorithm can bring the advantages of a non-linear modeling ability, a better generalization ability, automatic learning of feature representation, and a faster optimization speed. The following is a summary of the specific contributions:

- EGWO with the non-linear change of parameter a contributes to the balance between the exploration and exploitation capability.
- The introduction of the chaotic disturbance mechanism is conducive to search diversity.
- The candidate migration mechanism ensures the accuracy of the global optimal solution to strengthen the global convergence ability.
- The attacking mechanism guarantees a trade-off between the exploration and exploitation capabilities.
- The EGWO-MLP model is built to identify crop disease.

The remainder of this paper is organized as follows. Section 2 presents previous studies about GWO, MLP, and ptheir ractical applications. MLP, GWO, and the proposed enhanced gray wolf optimizer (EGWO) are introduced in Section 3. Section 4 explains the EGWO-MLP model. Section 5 presents the experiments and dataset. Section 6 analyzes and discusses the experimental results. The soybean identification model is expressed in Section 7. In Section 8, the conclusion of the paper and prospects are presented. The overall structure of the whole study is shown in Figure 3.

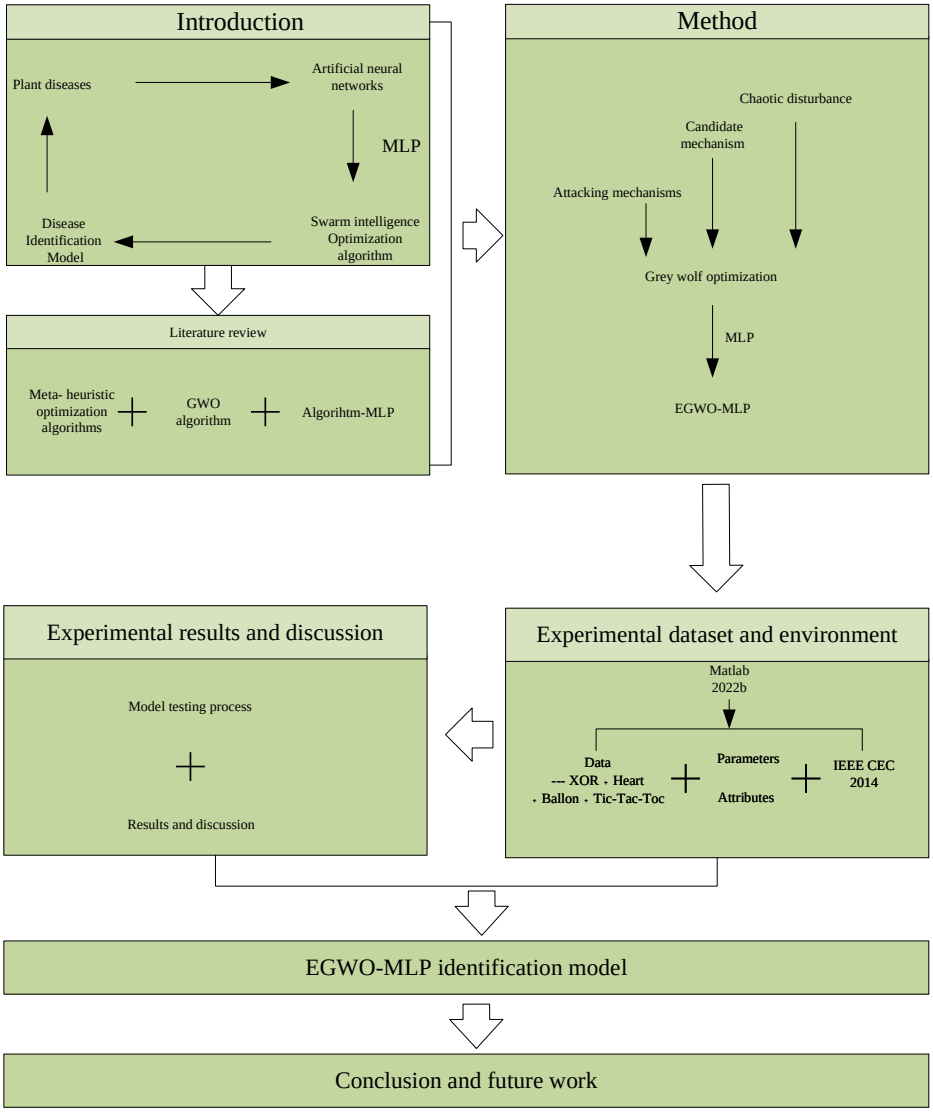


Figure 3. The overall structure of the whole study.

2. Literature Review

2.1. Meta heuristic Optimization Algorithms (MOAs)

Meta-heuristic optimization algorithms (MOAs) are a class of universal algorithms that can solve complex optimization problems. These algorithms do not rely on knowledge of specific problem domains but instead use highly exploratory search strategies to find global or near-optimal solutions in the solution space [34]. Tube element heuristic algorithms cannot guarantee finding the global optimal solutions, but they have been proven effective in achieving excellent results in many practical applications. In addition, compared with the traditional deterministic algorithm, meta-heuristic algorithms have the advantages of good parallel performance, strong global search ability, and adaptability to the problem structure. Computational techniques that derive inspiration from physical or biological logical phenomena can solve optimization problems. These can be divided into four categories: algorithms (a) based on physics, (b) based on evolution, (c) based on population, and (d) based on humans.

(1) Evolutionary computation class algorithms

This idea is mainly used to optimize the search space continuously by simulating biological evolution and genetic mechanisms [35,36]. Standard evolutionary computation algorithms include genetic algorithms, differential evolution algorithms, etc.

(2) Swarm intelligence optimization algorithm

This algorithm mainly simulates the behavior of certain natural groups, such as ant colonies, bird colonies, fish colonies, etc., treating the search space as an "ecosystem" to achieve a global optimal search through collaborative action. Common swarm intelligence algorithms include the Ant Algorithm (AA) [37], particle swarm optimization (PSO) [38], the Artificial Fish Swarm Algorithm (AFSA) [39], etc.

(3) Physics-based optimization algorithm

Physics-based meta-heuristic optimization algorithms are a class of heuristic algorithms inspired by physical phenomena and principles [40]. These algorithms simulate physical processes or physical behaviors in nature to solve optimization problems. Common physics-based meta-heuristic optimization algorithms are as follows: Simulated Annealing (SA) [41], the Gravitational Search Algorithm (GSA) [42], the Atomic Search Algorithm (ASO) [43], Big-Bang Big-Crunch (BBBC) [44], etc. These physics-based meta-heuristic optimization algorithms are flexible, easy to implement, and applicable to various optimization problems. They can be widely used in continuous, discrete, and combinatorial optimization, etc., with global search and fast convergence abilities to some degree.

(4) Human-based optimization algorithms

Human-based meta-heuristic optimization algorithms are a class of heuristic algorithms that are inspired by the way humans think and behave. These algorithms attempt to solve optimization problems by simulating human cognition, learning, and decision-making processes. Some of the human-based techniques include Forensic-Based Investigation Optimization (FBIO) [45], Political Optimizer (PO) [46], and Heap-Based Optimizer (HBO) [47]. These human-based meta-heuristic optimization algorithms are inspired by human intelligence and behavior with global search and fast convergence abilities to some degree. They can be widely used in continuous, discrete, and combinatorial optimization, and have achieved good results in solving complex problems.

2.2. Improved GWO and Its Application

The GWO algorithm is a meta-heuristic optimization algorithm based on group behavior inspired by the social behavior of gray wolves. This algorithm has been widely used and researched to solve various optimization problems. Since the GWO algorithm was proposed, it has been used in many fields and achieved remarkable results. The GWO algorithm is simple and easy to implement and adjust. The algorithm can find the global optimal solution in the search space by combining random search and the social behavior strategy. The GWO algorithm has been widely used in continuous, discrete, and combinatorial optimization problems. Researchers have made many improvements to and extensions of the GWO algorithm to improve its performance and adaptability. For example, introducing methods such as adaptive parameter control, chaotic search strategies, local search mechanisms, and multi-group structures enhances the convergence and searchability of the algorithm. The GWO algorithm has become an effective tool for solving optimization problems and has received extensive attention in both theoretical research and practical applications.

• Mechanism Innovation

To ensure a precise approximation to the global optimum, an algorithm named mGWO is proposed. The exploration and exploitation capabilities are balanced by adjusting the operator to improve the search accuracy for the global optimal solution [29]. Furthermore, to solve the premature convergence and local stagnation problems, a gray wolf optimizer (DSGWO) based on diversity-enhanced strategies, such as group competition mechanisms and exploration–exploitation balance mechanisms, was proposed to improve the performance of the GWO [48]. A collaboration-based hybrid optimizer called chHGWOSCA was introduced to balance exploration and mining capabilities. This optimizer combines the gray wolf optimization algorithm and the sine–cosine algorithm (SCA). It improves the parameter a to enhance global fusion to inspire inspiration and improve the global fusion

effect [49]. A hybrid GWO–sine cosine algorithm (SCA) (GWOSC) has been proposed [50]. GWO (gray wolf optimization) and the SCA (Sine Cosine Algorithm) are responsible for different tasks in different update stages to balance exploration and mining capabilities. Inspired by the gaze-cue behavior of wolves, two novel search strategies have been developed, including Neighbor Gaze-Cue Learning (NGCL) and Random Gaze-Cue Learning (RGCL) [51]. In addition, a weighted distance gray wolf optimizer (wdGWO) has been proposed, where the weighted sum of the best positions is used instead of just simple positions [52]. An improved algorithm called the exploration-enhanced GWO (EEGWO) algorithm, which improved exploration capabilities, was developed [53]. In addition, a nonlinear control parameter strategy is used to control the balance between the exploration and exploitation capability.

• Practical Application

To optimize the Radio Resource Management (RRM), a fractional GWO-CS optimization model integrating the GWO with the cuckoo search (CS) algorithm has been proposed, which optimizes parameters like the power spectral density (PSD), the transmission power, and the sensing bandwidth (SB) [54]. GWO has been combined with a gated recurrent unit (GRU) neural network to compensate for the reduced calibration time and ensure the fibre optic gyroscope's (FOG) calibration accuracy [55]. The GWO algorithm combining the Integer Wavelet Transform (IWT) reduces information loss regarding image steganography [56]. Then, the inverse of IWT and unsigncryption algorithms is utilized to extract the secret image from the stego image. A functional composition integration approach has been adopted to develop a hybrid meta-heuristic algorithm called HGWO-PSO [57]. HGWO-PSO combines the advantage of GWO algorithms and particle swarm optimization (PSO) and considers Clerc's parameter setting to improve decision making in the oil and gas industry. A hybrid GWO and differential evolution algorithm (HGWODE) is proposed by cooperating with the GWO and DE algorithm to balance exploration and exploitation [58].

• Feature Selection

A modified Gray Wolf optimizer is proposed by introducing the ReliefF algorithm and Coupla entropy in the initialization process to improve the quality of the initial population [59]. In addition, two new search strategies are adopted into the GWO to enhance the search more flexibly and avoid local stagnation [58]. The present study proposes a feature selection model for NIDS, which is based on particle swarm optimization (PSO), GWO, firefly optimization (FFA), and the genetic algorithm (GA) [60]. It aims to improve the performance of NIDS and shows promising results in terms of the false positive rate (FPR). Then, a binary version of a hybrid two-stage multi-objective FS method based on PSO and GWO is proposed. The first goal is to minimize the classification error rate, and the second is to reduce the number of selected features. The proposed FS method performs more efficiently and effectively than other meta-heuristic, statistical, and multi-objective FS methods [61]. A multi-strategy ensemble GWO (MEGWO) has been proposed. The proposed MEGWO incorporates three search strategies to update the solutions [62]. The present study proposes a binary hybrid GWO and Harris Hawks Optimization (HHO) to form a memetic approach called HBGWOHHO. Then, the continuous search space into a binary one to satisfy the feature selection based on the sigmoid function [63]. A binary method is developed into a two-phase multi-objective FS approach, based on PSO and GWO, which is applied to feature selection [64].

• Optimization of the Artificial Neural Network (ANN)

The GWO optimizes the weights of the networks to reduce the error. The GWO-ANN model is developed to improve the accuracy, which compares PSO, multiple linear regression (MLR), and nonlinear regression (NLR) models [65]. Two machine learning techniques have been adopted, i.e., ANN and GWO, to predict the level of road crash severity [66]. The proposed approach is a novel hybrid machine learning model that combines an ANN and the augmented gray wolf optimizer (AGWO). Based on the experimental findings, the

suggested ANN-AGWO can be utilized as a high-performance tool. The fuzzy C-means clustering algorithm (FCM) has been used to cluster historical electricity consumption data. At the same time, the FCM-GWO-BP neural network model is proposed to predict the energy consumption [67]. An improved GWO based on Levy flight has been proposed to help GWO jump out of local stagnation. Iit can be applied to train backpropagation (BP) [68].

• Algorithm Optimization of the MLP

An MLP has been used with a genetic algorithm (MLP-GA) to estimate the detection efficiency using metrics [24]. An electrical conductivity (EC) model was constructed based on the hybrid machine learning model of MLP-GWO. The hybrid MLP-GWO model has potential implications in precision agriculture [69]. GWO-MLP, PSO-MLP, and SSA-MLP have been proposed to be trained on different objective functions [70]. Two classes of algorithms, including bio-inspired and gradient-based algorithms, have been adopted to train the MLP for pattern classification [71]. An artificial immune network (opt-aiNet), PSO, and an evolutionary algorithm (EA) were used to train MLP networks. In addition, the standard backpropagation with momentum (BPM), a quasi-Newton method (DFP), and the modified scaled-conjugate gradient (SCGM) were used to train MLP networks [72]. The PSO algorithm was adopted to optimize the feedforward ANN. This method was designed to predict the maximum power point of a photovoltaic array, training feedforward neural networks (FNNs) with GSA, PSO, and GSA to alleviate local stagnation and accelerate convergence [73].

3. Method

3.1. Multi-Layer Perception

Multi-Layer perceptron (MLP) is also known as an ANN. It is a feed-forward structured artificial neural network that maps a set of input vectors to a group of output vectors, which can be seen in Figure 4. In addition to the input and output layers, MLP has multiple hidden layers in the middle. MLP is an extension of the perceptron, which overcomes the weakness that the perceptron cannot recognize linearly inseparable data. The simplest MLP has only one hidden layer, and a three-layer structure. MLP is fully connected with the input, hidden, and output layers, which have a hidden layer that can learn an arbitrary non-linear function of the input (with an infinite number of hidden nodes). MLP networks consist of multiple layers of neurons related to each other through directed connections, forming a directed graph-like structure. Each node fully connects with the next layer of nodes. Each node is a neuron with a non-linear activation function except for the input node. The MLP network is trained to improve its performance in supervised learning using the backpropagation algorithm.

An MLP is composed of multi-layer directional connected neurons [74], it has non-linear, high parallelism, anti-noise, fault-tolerance, and high generalization abilities to learn [75], and it has been widely used in many practical problems, such as nonlinear discriminants. If an MLP is used for regression, it can approximate the nonlinear input function. Furthermore, any function with continuous inputs and outputs can be approximated by an MLP. Novel optimization algorithms are suitable for training neural networks. These algorithms are based on sequential operator splitting techniques for specific related dynamical systems [76]. MLP training aims to find an optimal set of weights and bias parameters that minimize the mean squared error (*MSE*) value, which is essential for the optimization process [77]. In other words, an efficient MLP can be constructed to minimize a given error criterion by continuously adjusting and updating the weights and biases parameters.

The relationship of the MLP network layer can be expressed as $i \rightarrow j \rightarrow k$, where i acts as a subscript to the upper neuron or as an input node. j acts as a subscript to the current layer of neurons or to the hidden layer of neurons. k serves as a subscript to the next layer of neurons or the output layer of neurons.

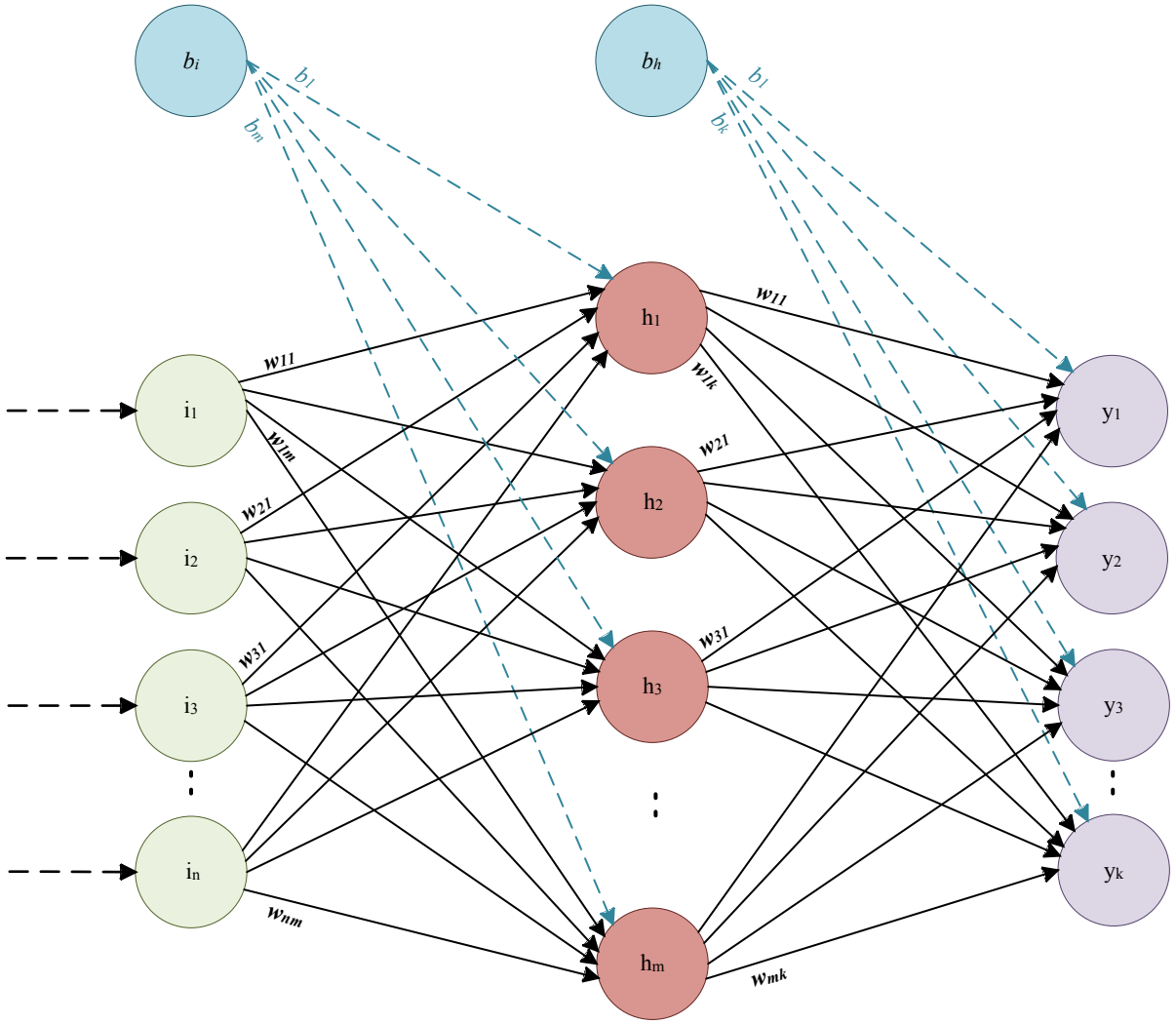


Figure 4. Architecture of a multi-Layer perceptron.

The weighted sum h can be computed by Equation (1).

$$h_j = \sum_{i=0}^{m} w_{ij} x_j \tag{1}$$

where w_{ij} represents the weight of each neuron in the previous layer to the current neuron, and w_{jk} represents the weight of the current neuron and the next layer of neurons, that is, the weight of the neuron k. h_j represents the sum of all input weights for the current node.

The output value of a hidden layer neuron is computed by Equation (2).

$$a_j = g(h_j) = g(\sum_{i=0}^{M}) w_{ij} x_{ij} \tag{2}$$

where a_j represents the output value of a hidden layer neuron. $g(h_j)$ represents an activation function, w is the weight, and x is the input. $a_j = x_{jk}$, i.e., the output value of the current neuron is equal to the input value of the next neuron. It can be computed by Equation (3).

$$y = a_k = g(h_k) = g \sum_{i=0}^{M} w_{jk} x_{ij} \tag{3}$$

where y denotes the value of the output layer, which is the final result. h_k represents the sum of the input weights of neurons in the output layer.

Each layer has an activation function. The sigmoid function is expressed by Equation (4).

$$g(h) = \sigma(h) = \frac{1}{1 + e^{-h}} \tag{4}$$

where the derivative of the sigmoid function can be computed by Equation (5).

$$\sigma'(x) = \sigma[1 - \sigma(x)] \tag{5}$$

To u the output layer weight w_{jk}, the gradient descent method can be used for the loss function, which can be expressed by Equation (6). w_{jk} can be obtained by Equation (7).

$$w_{jk} \leftarrow w_{jk} - \eta \frac{\partial E}{\partial w_{jk}} \tag{6}$$

$$w_{jk} = w_{jk} - \eta \sigma_o(k) a_i \tag{7}$$

where w_{jk} is the output value of the previous layer, that is, the input value of the output layer x_i. The hidden layer term can be updated by Equation (8).

$$\partial_h(j) = g'(h_j)(\sum_{k=1}^{N} \sigma_o(k) w_{jk}) \tag{8}$$

The hidden layer weights v_j (w_{ij}) can be updated by Equation (9).

$$v_j = v_j - \eta \; a_j(1 - \; a_j)(\sum_{k=1}^{N} \sigma_o(k) \; w_{jk}) \; a_i \tag{9}$$

where a_j is the output value of the current neuron, $a_j = g(h_j)$. a_j is the input value of the neuron in the current layer (the output value of the previous layer). When there are hidden layers with multiple layers, the weight can be computed by Equation (10).

$$v_j = v_j - \eta a_j(1 - a_j)(\sum_{k=q}^{N} \sigma_h(k) wjk) \; a_i \tag{10}$$

The final output can be computed by Equation (11).

$$p = \sum a_{ij} \; w_{ij} = \; h_j \tag{11}$$

3.2. Gray Wolf Optimizer

3.2.1. Principle of Motion

The gray wolf optimizer (GWO) is a new swarm-based algorithm that simulates the behavior and leadership roles in a sub-society of a pack of wolves and is inspired by the social behavior of gray wolves, such as the hierarchy and hunting mechanisms [26]. There are four types of wolves in a wolf pack: α wolf makes every decision and is responsible for the survival of all members of the pack; β wolves, with a social status in the pack after α wolf; δ wolves, which are responsible for caring among pups or all packs; and ω wolves, with the lowest social status in the pack. Gray wolves encircling their prey and marching during the hunting process are modeled using the above relationship. The basic GWO includes the following three main processes.

- To track and approach prey.
- To harass, chase, and surround prey until the prey stops moving.
- To attack the prey.

According to the division of the above levels, α wolves have absolute control over β, δ, and ω wolves; β wolves have absolute control over δ and ω wolves. δ wolves have absolute control over ω wolves. They play a key role in the main hunting process. Many ω wolves are usually in the best position to attack their prey. The whole optimization process can be divided into two stages:

The first stage is to surround the prey in response to its surrounding mechanisms (Equation (12))

$$D = |C \times X_p(t) - X(t)| \tag{12}$$

$$X(t+1) = X_p(t) - A \times D \tag{13}$$

where t is the current number of iterations. $x_p(t)$ is the position of the prey (equivalent to α, β, δ, ω wolves). r_1 and r_2 are random values in the range of [0, 1]. Furthermore, coefficient vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ are computed by Equations (14) and (15).

$$\overrightarrow{A} = 2\overrightarrow{a} \times \overrightarrow{r_1} - \overrightarrow{a} \tag{14}$$

$$\overrightarrow{C} = 2 \times \overrightarrow{r_2} \tag{15}$$

where $\overrightarrow{A}$ and $\overrightarrow{C}$ are condition vectors. $\overrightarrow{X}$ represents the position vector of the wolf. Furthermore, $\overrightarrow{A}$ is a random value in the range $-2\overrightarrow{a}$ and $2\overrightarrow{a}$. The iterative process randomly selects r_1 and r_2 in the normal range of 0 to 1. The component of $\overrightarrow{a}$ decreases linearly from 2 to 0 by Equation (16).

$$\overrightarrow{a} = 2 - l * \frac{2}{MaxIter} \tag{16}$$

where l is the current iteration and *MaxIter* is the max iterations.

In the search process, the position migration of the α, β, δ wolves is calculated according to Equations (17)–(19).

$$\overrightarrow{D} = |\overrightarrow{C_1} \times \overrightarrow{X_a} - \overrightarrow{X}| \tag{17}$$

$$\overrightarrow{D_\beta} = |\overrightarrow{C_2} \times \overrightarrow{X_\beta} - \overrightarrow{X}| \tag{18}$$

$$\overrightarrow{D\delta} = |\overrightarrow{C_3} \times \overrightarrow{X_\delta}| \tag{19}$$

The second stage: The α wolf dominates the whole process, while the β and δ wolves are also involved. When hunting, they lead ω wolves to update their position, and other wolves move randomly when looking for prey. Equations (20)–(24) measure the location of the prey and search around the prey until they finally find it. During this process, they always maintain a high level of coordination and cooperation to ensure the successful capture of the prey.

$$D_\delta = |C_\delta \times X_\delta - X_i| \tag{20}$$

$$X_1 = X_\alpha - A_\alpha \times (D_\alpha) \tag{21}$$

$$X_2 = X_\beta - A_\beta \times (D_\beta) \tag{22}$$

$$X_3 = X_\delta - A_\delta \times (D_\delta) \tag{23}$$

$$X(t+1) = \frac{x_1 + x_2 + x_3}{3} \tag{24}$$

The final position is anywhere inside the circle. The primary goal of the ω wolf is to update their position according to α, β, and δ wolves.

3.2.2. Insufficiency of the Algorithm

The GWO differs from other optimization algorithms in that it draws inspiration from the social and predation behaviors of gray wolves and simulates the cooperation and competition mechanisms of gray wolf groups. It employs specific search strategies for information exchange and knowledge sharing through direct communication. In contrast, other optimization algorithms may have different sources of inspiration, search strategies, and communication methods. In addition, the parameter settings of the GWO are relatively simple and one only needs to determine the initial population size and the upper and lower limits of the search range. However, the GWO also has some shortcomings. First of all, it is sensitive to the selection of the initial solution, and the quality of the initial solution may affect the final optimization performance of the algorithm. Second, the convergence speed of GWO may be fast in some cases, but it may also be unstable, leading to a locally optimal solution. In addition, for high-dimensional problems, GWO faces challenges because the gray wolf behavior simulation may need to be adapted to searches in high-dimensional spaces. Although the GWO has unique characteristics and advantages, its shortcomings must also be considered. In practical applications, selecting a suitable optimization algorithm according to specific problems and requirements is necessary.

3.3. The Proposed Enhanced Gray Wolf Optimizer Algorithm (EGWO)

Although the GWO takes advantage of the parameters to strike a balance between exploration and exploitation, it still needs to solve the problem of suboptimal solution stagnation and premature convergence, leading to the algorithm's slow convergence. In the GWO algorithm, α, β, and δ wolves guide ω wolves to attack prey, where it is assumed that α wolves are in the best position for the prey position. During the search, α, β, and δ wolves are selected from the population, while the remaining wolves are treated as ω wolves and can be relocated to improve algorithm performance. However, this mechanism has some defects, quickly leading to premature convergence and local stagnation of the algorithm. This paper proposes an improved version of the GWO algorithm named enhanced gray wolf optimization algorithm (EGWO) to solve the above problem. Although the GWO takes advantage of parameters to strike a balance between exploration and exploitation, it still needs to solve the problem of suboptimal solution stagnation and premature convergence, leading to the algorithm's slow progress.

First, to improve the parameter a, change the parameter a from a linear change to a non-linear change, to achieve a balance between exploration and exploitation. Parameter a is calculated by Equation (25).

$$a = 2 \times e^{-(2 \times \frac{t}{max-iter})} \tag{25}$$

3.3.1. Chaotic Disturbance

Chaotic disturbance can effectively ensure the global diversity of the algorithm and enhance its exploration ability. Iterative mapping is added to the algorithm. Iterative mapping refers to approaching a certain target by repeatedly applying the mapping function until certain conditions are met, or a certain convergence is achieved. We can add adaptability and optimization capabilities to the algorithm by introducing iterative mapping to handle complex problems better. According to Figure 5, as the number of iterations increases or decreases, it also follows an irregular change. This change can effectively update the size and direction of wolf steps, avoiding falling into local optima during migration. The chaotic coefficient is calculated using Equations (26) and (27).

$$k(t+1) = sin(\frac{0.7 \times \pi}{k(t)}) \tag{26}$$

$$G(t) = \frac{(k(t)+1) \times 100}{2} \tag{27}$$

where $k(t)$ is the parameter under the t-th iteration and the $G(t)$ is the chaos mapping parameter.

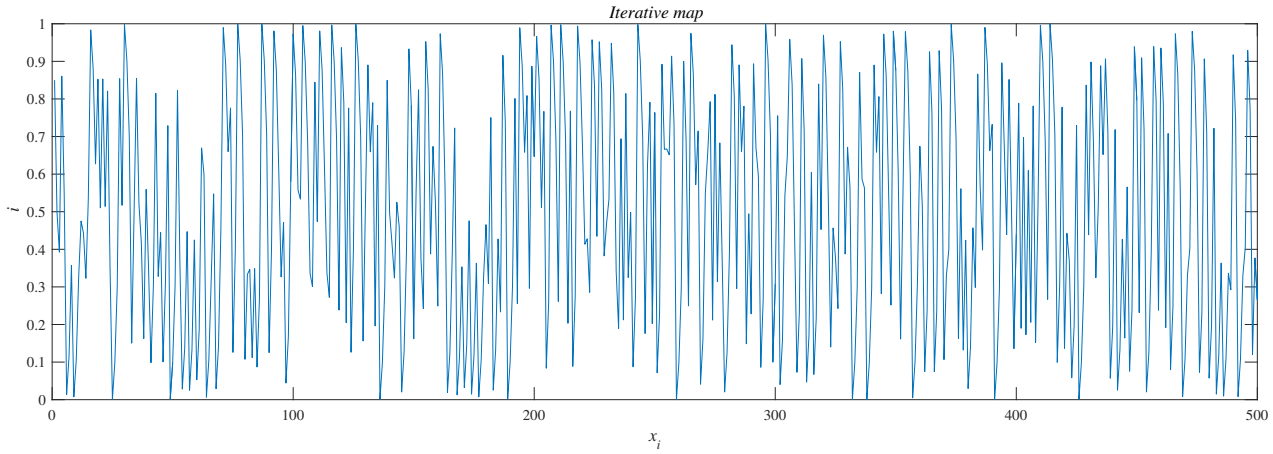


Figure 5. The chaotic map update with iteration number.

3.3.2. Candidate Migration Mechanism

In the GWO algorithm, there is a situation where the optimal solution is lost, and the accuracy of the optimal solution is not guaranteed. A candidate mechanism is introduced computing by Equations (28) and (29). During the candidate mechanism, the center points of the three wolves are added to construct a candidate tribe pool based on the three wolves and the center positions of the the three wolves. The above mechanism can ensure that the optimal solution is not lost while avoiding local stagnation, promoting the possibility of local stagnation avoidance during the update process.

$$Cand = \frac{(X_\alpha + X_\beta + X_\delta)}{3} \tag{28}$$

$$CandPool = [X_\alpha, X_\beta, X_\delta, Cand] \tag{29}$$

where $Cand$ is the position of the three wolves, $CandPool$ is the candidate pool with the α, β, and δ wolves and $Cand$ wolf. α, β, and δ wolves play a crucial role in the wolf pack migration. Equations (30)–(35) can update the position migration.

$$D_\alpha = |a \times X_\alpha - X_i| \times (G(t) \times \frac{(rand - 0.5)}{10}) \tag{30}$$

$$X_1 = CandPool_{r,j} - A_1 \times D_\alpha \tag{31}$$

$$D_\beta = |a \times X_\beta - X_i| \times (G(t) \times \frac{(rand - 0.5)}{10}) \tag{32}$$

$$X_2 = CandPool_{r,j} - A_1 \times D_\beta \tag{33}$$

$$D_\delta = |a \times X_\delta - X_i| \times (G(t) \times \frac{(rand - 0.5)}{10}) \tag{34}$$

$$X_3 = CandPool_{r,j} - A_1 \times D_\delta \tag{35}$$

where $CandPool_{r,j}$ is the random wolf from the candidate pool. $G(t)$ is the mapped parameter based on the chaotic map. The direction parameter $(G(t) \times (rand - 0.5)/10)$ can determine the update direction and step length.

3.3.3. Attacking Mechanism

As the iteration updates, the attack mechanism also changes accordingly. Different attack methods can enhance the diversity of wolf packs, and two other attack methods can effectively improve the algorithm's exploration ability. When the update phase is less $(1/2 \times maxiteration)$, select Equation (36) as the attacking method, and when the update phase is more $(1/2 \times maxiteration)$, the attacking method can be computed by Equation (37).

$$W_{i,j} = W_1 \times X_1 + W_2 \times X_2 + W_3 \times X_3 \tag{36}$$

$$X_{i,j} = \frac{X_1 + X_2 + X_3}{3} \tag{37}$$

where F is the total fitness based on the three wolves by Equation (38), w_1, w_2, and w_3 are the weights computed by Equation (39), and $X_{i,j}$ is the attacking position.

$$F = F_\alpha + F_\beta + F_\delta \tag{38}$$

$$W_1 = \frac{F_\alpha}{F}; W_2 = \frac{F_\beta}{F}; W_3 = \frac{F_\delta}{F} \tag{39}$$

where F_α, F_β, and F_δ are the fitness of the α, β, and δ wolves.

Based on the above mechanism, we propose a new algorithm named EGWO. In the original GWO algorithm, α wolf is the starting point for random initialization and weights, which can converge to ω wolf but also tend to fall into local optimal solutions. By introducing the new mechanism, the ability of wolves in global and local search is demonstrated. To summarize, the mechanism advantages can be expressed as follows:

Firstly, the non-linear change in parameter a can balance the exploration and exploitation capability.

Secondly, the chaotic disturbance mechanism can manage the step direction and length, which contributes to search diversity.

Thirdly, the candidate migration mechanism updates the position of three wolves to promote the search to jump out of local stagnation, ensure the accuracy of the global optimal solution, and strengthen the global convergence ability.

Fourth, introducing an attacking mechanism can effectively strengthen the exploration capability and ensure a balance with the exploration capability.

3.4. Computational Complexity of EGWO Algorithm

The computational complexity of the EGWO algorithm is described through two aspects: time complexity and space complexity. The above aspects are important factors in evaluating the performance of an algorithm.

(1) Time complexity

The number of particles (N), the number of iterations (t), and the cost of function evaluation (c) are the important factors affecting the time complexity. We must fully integrate their effects to obtain an accurate time complexity evaluation. It can be seen that the time complexity of the EGWO is equal to the GWO, which is maintained constant by Equations (40) and (41).

$$O(GWO) = O(tNc) \tag{40}$$

$$O(EGWO) = O(tNc) \tag{41}$$

(2) Space complexity

For space complexity, only the initial stage, i.e., the entire search space, is considered. Then, the space complexity of the EGWO is $O(n)$.

4. Combing EGWO with Multi-Layer Perceptron (EGWO-MLP)

4.1. EGWO-MLP Optimization Model

The training process is divided into four steps: preprocessing, learning, evaluation, and prediction. The first step is preprocessing. Firstly, the data are preprocessed and handled for better use in subsequent modeling and analysis. Secondly, the data are divided into training data and testing data. The second step is the learning process. The optimization algorithm continuously optimizes MLP to avoid local stagnation in the modeling process. The third step is evaluating the obtained model using evaluation criteria such as *MSE*. The final step is to predict the results and the final experimental results. The identification process can be seen in Figure 6.

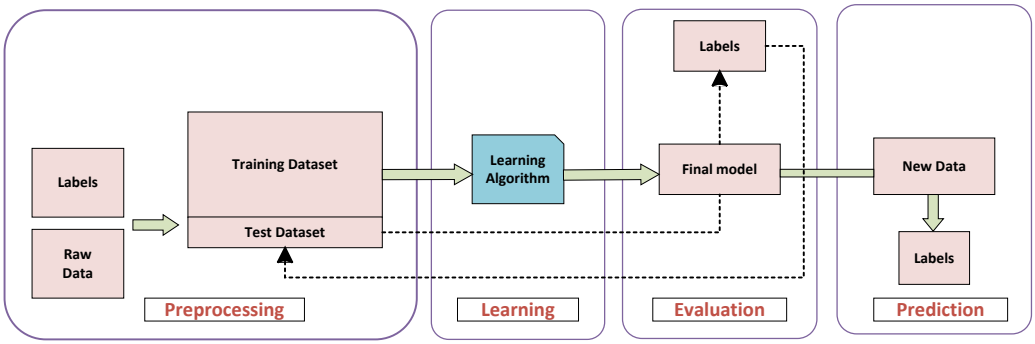


Figure 6. Training the whole identification model process.

The purpose of training a network via the proposed EGWO algorithm is to determine the weights and biases. The obtained weights and biases compute the expected network's output value when the network presents different inputs. The EGWO-MLP identification model aims to identify plant disease types. Hidden layer structure features and dynamic weight parameter adjustment make it more accurate in identifying the disease types. The EGWO-MLP identification model structure consists of input, hidden, and output layers. The process includes normalization processing, input determination, output, hidden units, training parameter settings, network model creation, activation function calls, etc. The output is the identification result. If the output of the test samples satisfies the training sample's expectation, the learning ends. The overall process is depicted in Figure 7, which can be summarized in five steps.

Step 1: Initialization stage of wolf pack. N wolves can be generated by the proposed EGWO algorithm.

Step 2: Weights and biases mapping phase. The solution (position) of the generated wolves via the EGWO algorithm is allocated as weights and biases.

Step 3: Update location phase. α, β, and δ wolves are computed by Equations (30)–(35).

Step 4: Evaluation phase. Evaluate the performance of EGWO algorithm in training the MLP network using *MSE* standards.

Step 5: Iterative update phase. The EGWO algorithm continuously updates the weight and bias terms of the MLP network until the termination condition is reached.

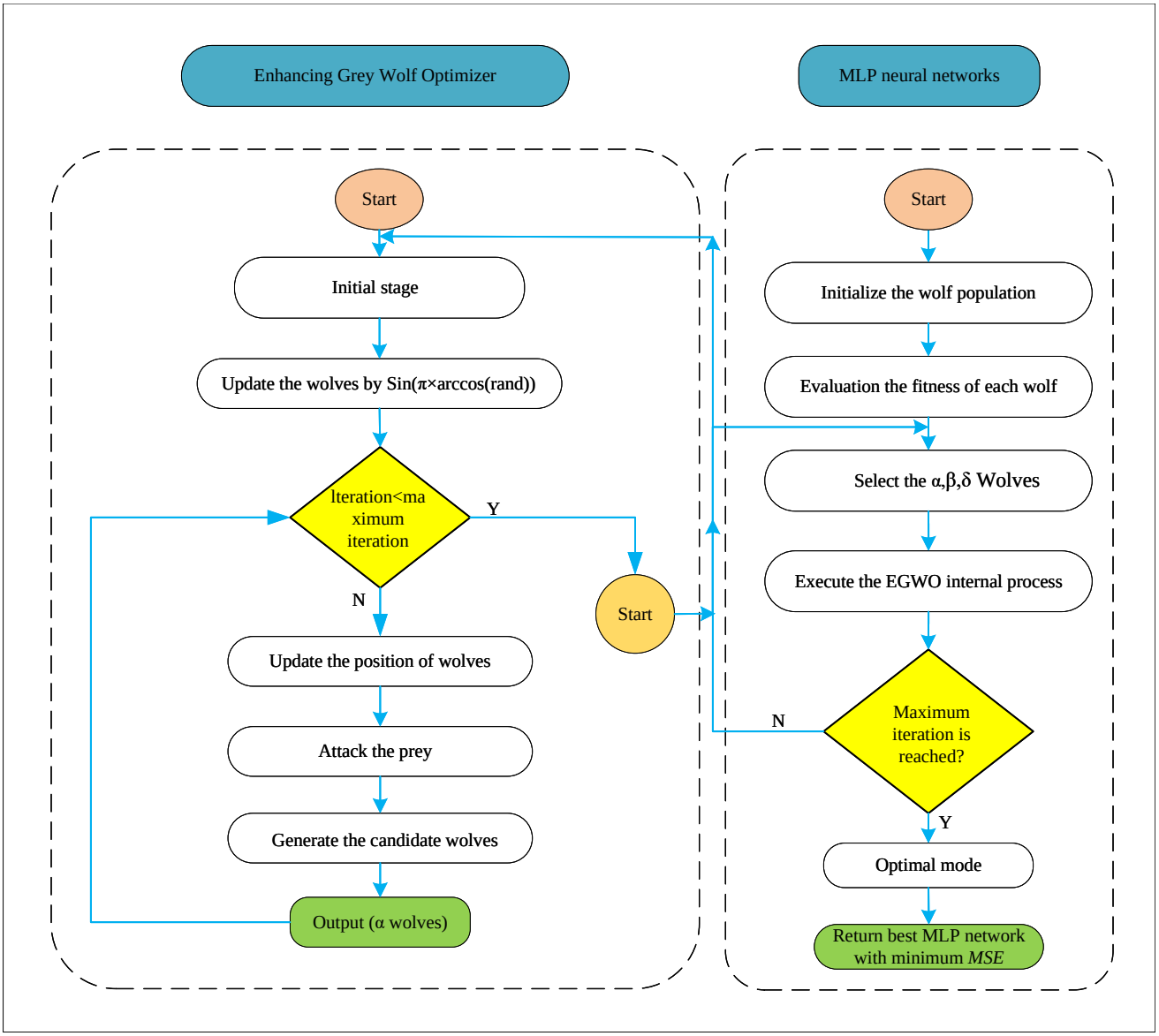


Figure 7. The process of the EGWO to train MLP.

4.2. Encoding Mechanism

The weights and biases of the MLP are obtained by the proposed EGWO algorithm, which is the best position (α wolf). The parameter (α wolf) can train the MLP network based on continuous updating and iterating. The position can be mapped as $\theta = I_w, h_w, h_b, O_b$, where I_w means the weights of the input nodes and h_w are the weights of the hidden nodes. Furthermore, h_b represents the biases of the hidden layer and O_b are the biases of the output layer. The wolf can be encoded as the weights and biases shown in Figure 8.

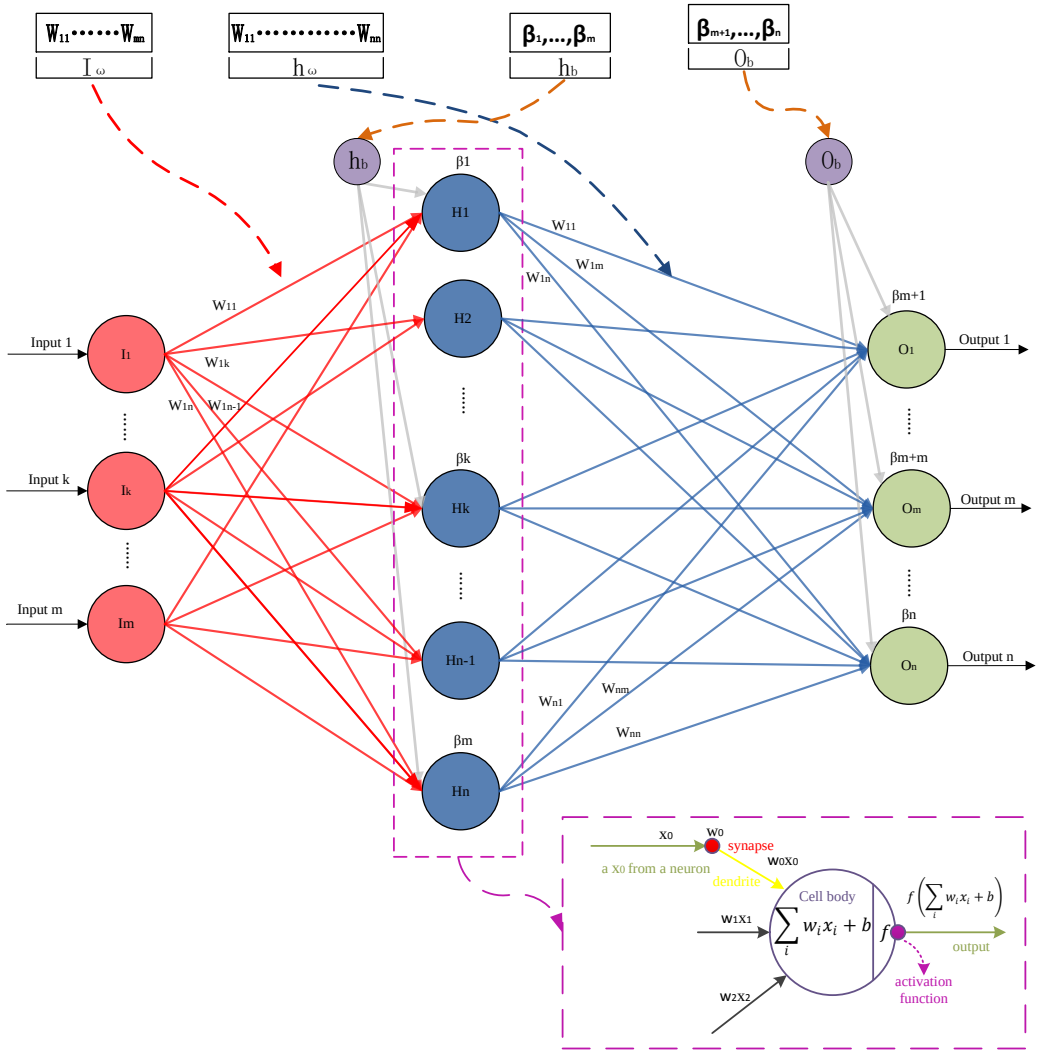


Figure 8. Mapping an EGWO solution to the MLP network.

4.3. Evaluation Criteria

The mean square error (*MSE*) is the loss function commonly used in regression tasks. The statistical parameter is the value of the sum of squared errors at the corresponding points of the predicted data and the original data. The difference between the actual and expected values is the criterion for evaluating the training algorithm. The *MSE* is widely used in tasks such as linear and multiple linear regression. In deep learning, the *MSE* is also used to measure the performance of neural networks in regression tasks and as a loss function for optimization. When using the *MSE* as a loss function for optimization, optimization algorithms such as gradient descent are usually used to minimize the value of the *MSE* and thus improve the model's performance. The smaller the difference, the better the algorithm trained and the closer the expected value is to the actual value. The *MSE* is a standard metric for evaluating MLPs; it is widely used and calculated by Equation (42).

$$MSE = \sum_{i=1}^{m} (o_i^k - d_i^k)^2 \tag{42}$$

where m is the number of outputs and d_i^k and o_i^k are the expected and actual outputs for ith inputs using k-th training samples. The MLP performance is evaluated by the average $\overline{MSE}$ of all training samples to be effective on the entire set of training samples, t. $\overline{MSE}$ is the average *MSE* calculated by Equation (43).

$$\overline{MSE} = \sum_{k=1}^{s} \frac{\sum_{i=1}^{m} (o_u^k - d_i^k)^2}{s} \tag{43}$$

Training an MLP model requires considering the effects of several variables and functions, and in the EGWO algorithm, the $\overline{MSE}$ is calculated by Equation (44). This equation involves the number of training samples s and requires consideration of several factors, including the network structure and parameter tuning. In order to obtain better training results, we need to reasonably adjust and optimize these factors to minimize the *MSE* and improve the generalization ability of the model.

$$minimize : F(\overrightarrow{V}) = \overline{MSE} \tag{44}$$

In addition to using the *MSE* to measure model performance, the classification accuracy can also be used to evaluate model performance. For categorical datasets, during model training, we need to plot the sample data by Equation (45) and classify them according to different categories. The accuracy metric's advantage is that it provides a more intuitive picture of the model's classification ability and helps us better understand its performance and optimization direction. Therefore, when training and optimizing the model, we need to consider both metrics together for a more comprehensive evaluation result.

$$Accuracy \quad rate = \frac{Number \quad of \quad correctly \quad classified \quad objects}{Number \quad of \quad objects \quad in \quad the \quad dataset} \tag{45}$$

4.4. Selection of Activation Function

This paper selects the activation function of training MLP as the *Sigmoid* function. It has an exponential function shape, closest to biological neurons in a physical sense, and is a standard S-type function in biology, also known as an *S-type* growth curve in Figure 9. In addition, the output of (0, 1) can also be expressed as a probability or used for normalization of the input. The *Sigmoid* function is a commonly used function in machine learning [21], and it is the most widely used type of activation function, which is expressed in Equation (46).

$$Sigmoid = \frac{1}{1 + e^{-z}} \tag{46}$$

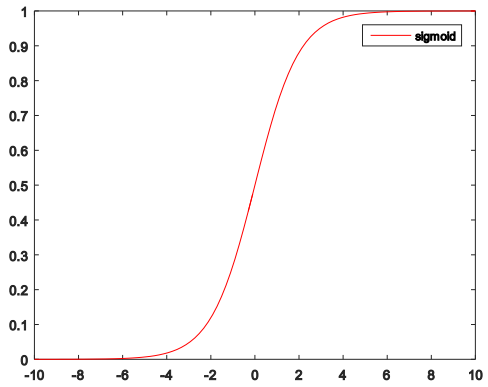


Figure 9. Sigmoid function S-shaped growth curve image.

The reasons why the *Sigmoid* function is widely used are summarized as follows:

- Its derivative reduces decay and dilution errors, signal problems, oscillation problems, and asymmetrical input problems. When this function is used to solve problems, it can be used for category classification and is suitable for prediction [78].
- The segmented linear recursive approximation method calculates the *Sigmoid* function and its derivatives in artificial neurons. This method helps the neuron to estimate the

Sigmoid function and its derivatives more accurately during the learning process so that the neuron can better process the input data and output the correct results [79].

5. Experimental Preparation

5.1. Experimental Setting

- Experimental environment. The experiment codes are executed in Matlab R2015b under the Windows 10 operating system, all simulations were run on a computer with an Intel (R) Core(TM) i5-9300 CPU @ 2.40 GHz, and its memory is 8G. Thirty runs for each working access the predictive performances. The population is set to 30, and the max iteration is 500 for the IEEE CEC 2014 benchmark functions and 100 iterations for the UCI dataset to verify the EGWO and EGWO-MLP.
- Data processing. In order to eliminate the dimensional impact between indicators, data are standardized to achieve comparability between data indicators. After the original data are standardized, all indicators are on the same order of magnitude so that comprehensive processing can occur. The experiment in this paper will process the data to the range of [0, 1], which uses the method of Min-Max normalization. Min-Max normalization can be computed by Equation (47).

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{47}$$

where x_{max} and x_{min} are the max and min value of the x current data values, respectively. x' is the standardized value.

5.2. Comparison Algorithm Selection

The EGWO algorithm is compared with other MLP algorithms, and the algorithm's ability for disease identification is verified. All parameters of the comparative algorithms are set as shown in Table 1.

Table 1. The initial parameter settings for the corresponding algorithms.

Algorithms	Parameters	Values
	GWO variants	
WdGWO	Null	Null
IGWO	Null	Null
GNHGWO	Null	Null
	Traditional and popular algorithms	
PSO	Coefficient of the cognitive component	2
	Coefficient of the social component	2
DE	Scale factor primary (F)	0.6
	Crossover rate (Cr)	0.8
BA	Loudness (A)	0.5
	Pulse rate (a)	0.5
	Frequency minimum	0
	Frequency maximum	2
TSA	ST	0.1
	The number of seeds (ns)	[0.1 × N, 0.25 × N]
SCA	a	2
	r_1	Linearly decreased from a to 0
BOA	p	0.8
	power exponent	0.1
	sensory modality	0.01
JAYA	Null	Null
	Traditional and popular algorithms	
SWO	TR	0.3
	Cr	0.02
	Minimum population size	20
ZOA	Null	Null
COA	Null	Null
BOA	Null	Null
OOA	Null	Null
CO	search agents in a group	2

5.2.1. GWO Variant

The variant algorithm refers to making improvements or modifications based on the original algorithm to achieve a better performance. By comparing different variant algorithms, we can obtain more choice and diversity to find the most suitable algorithm for a specific problem. Therefore, GWO variants are selected for performance comparison, and their performance can be evaluated more comprehensively by comparing different variant algorithms.

- Improved gray wolf optimization (IGWO) introduces an adaptive weight mechanism, which can dynamically adjust an individual's weight according to its fitness value so that individuals with a higher fitness have more significant influence [80]. Through this mechanism, the IGWO algorithm can more effectively explore the search space and speed up the convergence. It has many applications for solving complex optimization problems, parameter optimization, feature selection, etc.
- Greedy non-hierarchical gray wolf optimizer (G-NHGWO) introduces a greedy strategy to increase the locality of the search [81]. In addition, the method also adopts a non-hierarchical optimization strategy, which avoids the use of fixed weight factors. G-NHGWO can search for the best solution more efficiently in practical problems and provide more accurate and reliable optimization results.
- Weighted distance gray wolf optimizer (WdGWO) introduces the concept of a weighted distance, which measures how close an individual wolf is to the current global best solution [52]. The WdGWO algorithm exploits the notion of social hierarchy among gray wolves to guide the search for promising regions of the solution space.

5.2.2. Traditional Algorithms

Traditional optimization algorithms have been widely used in research and have good performance and effects in many problem areas. Traditional optimization algorithms have a high interpretability and reliability.

- Particle swarm optimization (PSO) has a more vital ability to explore the solution set space for non-convex optimization problems. It is relatively simple, and the calculation process is separated from the problem model. As a population-based meta-heuristic algorithm, PSO is applicable to distributed computing and can effectively improve the computing power. Its speed (update) mechanism, inertia, and other factors can be well optimized for parameter optimization in ANNs [38]
- Differential evolution (DE) is a heuristic random search algorithm based on population differences. The differential evolution algorithm has the advantages of simple principles, few controlled parameters, and strong robustness [82].
- The Bat Algorithm (BA) is an optimization algorithm for simulating bat swarm behavior, which has multiple advantages such as parallelism, an adaptive search strategy, diversity maintenance, a relatively simple implementation, powerful global search capability, and adaptability. Its adaptability can adjust the search strategy according to the characteristics of the problem and improve the robustness and global search ability. Randomness and exploration operations are introduced to maintain population diversity, avoid falling into local optimal solutions, and provide more comprehensive search space coverage [83].
- The Tree-seed Algorithm (TSA) has a simpler structure, a higher search accuracy, and a stronger robustness than some traditional intelligent optimization algorithms [84].
- The Sine-Cosine optimization algorithm (SCA) is a random optimization algorithm that is highly flexible, simple in principle, and easy to implement. It can be easily applied to optimization problems in different fields [85].
- The Butterfly Optimization Algorithm (BOA) solves global optimization problems by mimicking the food searching and mating behavior of butterflies [86]. The design framework of the algorithm is mainly based on the foraging strategy of butterflies looking for nectar or mating partners, in which butterflies use their sense of smell to

determine the target location. The BOA algorithm draws on this foraging behavior and combines the characteristics of optimization algorithms to provide an effective solution for complex global optimization problems.

5.2.3. Recent Algorithms

Over time, new algorithms emerge, often with more advanced techniques and better performance. Using recently emerged algorithms as benchmarks for comparative experiments can provide more accurate and unbiased evaluation criteria. By comparing with the latest algorithm, the advantages and disadvantages of the proposed algorithm can be evaluated more objectively.

- The Spider Wasp Optimizer (SWO) is inspired by the hunting, nesting, and mating behavior of female spider wasps in nature [87]. Furthermore, it shows promising results in solving various optimization problems with different exploration and development requirements through various unique update strategies.
- The Zebra Optimization Algorithm (ZOA) is a heuristic optimization algorithm that simulates the behavior of zebra swarms and is used to solve optimization problems [88]. The ZOA algorithm searches for the optimal solution in the solution space by imitating the foraging and migration strategies of the zebra population. The core idea of the ZOA is to regard the candidate solution of the problem as an individual zebra and search by simulating the foraging and migration behavior of zebra.
- The Reptile Search Algorithm (RSA) is inspired by the hunting behavior of crocodiles [89]. The implementation of the algorithm includes two key steps: encirclement and hunting. This makes the RSA algorithm adaptable to different optimization problems and have better exploration and exploitation capabilities.
- The Brown-bear Optimization Algorithm (BOA) is inspired by the communication patterns of pedal scent marking and sniffing behavior of brown bears, and utilizes the communication behavior characteristics of brown bears and provides an effective solution by simulating their strategies in finding food and marking territory [90].
- The Osprey Optimization Algorithm (OOA) mimics the behavior of ospreys in nature and is mainly inspired by an osprey's strategy when fishing at sea [91]. Ospreys detect the location of their prey, hunt it down, and bring it to a suitable place to enjoy it. OOA algorithms can efficiently solve various optimization problems and balance exploration and exploitation during the search process.
- The Cheetah Optimizer (CO) is proposed by simulating cheetahs' hunting behavior and related strategies. The cheetah optimizer can effectively solve various optimization problems and adapt to complex environments [92].

5.3. Standard Test Set

5.3.1. IEEE CEC 2014 Benchmark Functions

The IEEE CEC 2014 benchmark functions are a benchmark test set for evaluating the performance of optimization algorithms [93] provided by the 2014 IEEE Congress on Evolutionary Computation (CEC) competition, which contains a total of 30 standard continuous optimization problems shown in Tables 2–5, covering many different types of functions. The IEEE CEC 2014 benchmark functions are designed to evaluate optimization algorithms' global search abilities, convergence speeds, accuracies, and robustness. It is used to compare the performance of different algorithms and improve and optimize their optimization algorithms. The IEEE CEC 2014 benchmark functions have become one of the standard benchmarks for evaluating the performance of optimization algorithms. It provides a fair and reliable platform for researchers to compare and verify the capabilities and effects of optimization algorithms.

Table 2. The unimodal benchmark functions of IEEE CEC 2014 benchmark functions.

Name of the Functions	Expression
Rotated High Conditioned Elliptic Function	$F_1(x) = f_1(M(x - o_1)) + 100$
Rotated Bent Cigar Function	$F_2(x) = f_2(M(x - o_2)) + 200$
Rotated Discus Function	$F_3(x) = f_3(M(x - o_3)) + 300$

Table 3. The multimodal benchmark functions of IEEE CEC 2014 benchmark functions.

Name of the Functions	Expression
Shifted and Rotated Rosenbrock's Function	$F_4(x) = f_4(M(\frac{2.048(x-o_4)}{100}) + 1) + 400$
Shifted and Rotated Ackley's Function	$F_5(x) = f_5(M(x - o_5)) + 500$
Shifted and Rotated Weierstrass Function	$F_6(x) = f_6(M(\frac{0.5(x-o_6)}{100})) + 600$
Shifted and Rotated Griewank's Function	$F_7(x) = f_7(M(\frac{600(x-o_7)}{100})) + 700$
Shifted Rastrigin's Function	$F_8(x) = f_8(M(\frac{5.12(x-o_8)}{100})) + 800$
Shifted and Rotated Rastrigin's Function	$F_9(x) = f_8(M(\frac{5.12(x-o_9)}{100})) + 900$
Shifted Schwefel's Function	$F_{10}(x) = f_9(M(\frac{1000(x-o_{10})}{100})) + 1000$
Shifted and Rotated Schwefel's Function	$F_{11}(x) = f_9(M(\frac{1000(x-o_{11})}{100})) + 1100$
Shifted and Rotated Katsuura Function	$F_{12}(x) = f_{10}(M(\frac{5(x-o_{12})}{100})) + 1200$
Shifted and Rotated HappyCat Function	$F_{13}(x) = f_{11}(M(\frac{5(x-o_{13})}{100})) + 1300$
Shifted and Rotated HGBat Function	$F_{14}(x) = f_{12}(M(\frac{5(x-o_{14})}{100})) + 1400$
Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	$F_{15}(x) = f_{13}(M(\frac{5(x-o_{15})}{100}) + 1) + 1500$
Shifted and Rotated Expanded Scaffer's F6 Function	$F_{16}(x) = f_{14}(M((x - o_{16}))1) + 1600$

Table 4. The hybrid benchmark functions of IEEE CEC 2014 benchmark functions.

Name of the Functions	Expression
$F_{17} = f_9(M_1Z_1) + f_8(M_2Z_2) + f_3(M_3Z_3) + 1700$	p = [0.3, 0.3, 0.4]
$F_{18} = f_2(M_1Z_1) + f_8(M_2Z_2) + f_3(M_3Z_3) + 1800$	p = [0.3, 0.3, 0.4]
$F_{19} = f_7(M_1Z_1) + f_8(M_2Z_2) + f_3(M_3Z_3) + f_8(M_4Z_4) + 1900$	p = [0.2, 0.2, 0.3, 0.3]
$F_{20} = f_{12}(M_1Z_1) + f_3(M_2Z_2) + f_{13}(M_3Z_3) + f_8(M_4Z_4) + 2000$	p = [0.2, 0.2, 0.3, 0.3]
$F_{21} = f_{14}(M_1Z_1) + f_{12}(M_2Z_2) + f_4(M_3Z_3) + f_9(M_4Z_4) + f_1(M_5Z_5) + 2100$	p = [0.1, 0.2, 0.2, 0.2, 0.3]
$F_{22} = f_{10}(M_1Z_1) + f_{11}(M_2Z_2) + f_{13}(M_3Z_3) + f_9(M_4Z_4) + f_5(M_5Z_5) + 2200$	p = [0.1, 0.2, 0.2, 0.2, 0.3]
Notes:	
$Z_1 = [y_{s_1}, y_{s_1} \ldots, y_{s_{n1}}]$	
$Z_2 = [y_{s_{n1+1}}, y_{s_{n1+2}} \ldots, y_{s_{n1+n2}}]$	
$Z_N = [y_{s_{\sum_{i=1}^{N-1} n1+1}}, y_{s_{\sum_{i=1}^{N-1} n1+2}} \ldots, y_{s_{5D}}]$	
$y = x - o_i$, $S = randperm(1:D)$, $percentageof g_i(x)$	
$n_1 = [p_1 D]$, $n_2 = [p_2 D]$,..., $n_{N-1} = [p_{N-1} D]$, $n_N = D - \sum_{i=1}^{N-1} n + i$	

Table 5. The composite benchmark functions of IEEE CEC 2014 benchmark functions.

Name of the Functions	Expression
$F_{23} = w_1 * F'_4(x) + w_2 * [1e^{-6}F'_1(x) + 100] + w_3 * [1e^{-26}F'_2(x) + 200]$ $+w_4 * [1e^{-6}F'_3(x) + 300] + w_5 * [1e^{-6}F'_1(x) + 400] + 2300$	$\sigma = [10, 20, 30, 40, 50]$
$F_{24} = w_1 * F'_{10}(x) + w_2 * [F'_9(x) + 100] + w_3 * [F'_{14}(x) + 200] + 2400$	$\sigma = [20, 20, 20]$
$F_{25} = w_1 * 0.25F'_{11}(x) + w_2 * [F'_9(x) + 100] + w_3 * [1e^{-7}F'_1(x) + 200] + 2500$	$\sigma = [10, 30, 50]$
$F_{26} = w_1 * 0.25F'_{11}(x) + w_2 * [F'_{13}(x) + 100] + w_3 * [1e^{-7}F'_1(x) + 200]$ $+w_4 * [2.5F'_6(x) + 300] + w_5 * [1e^{-6}F'_{13}(x) + 400] + 2700$	$\sigma = [10, 10, 10, 10, 10]$
$F_{27} = w_1 * 10F'_{14}(x) + w_2 * [10F'_9(x) + 100] + w_3 * [2.5F'_1 1(x) + 200]$ $+w_4 * [25F'_{16}(x) + 300] + w_5 * [1e^{-6}F'_1(x) + 400] + 2700$	$\sigma = [10, 10, 10, 20, 20]$
$F_{28} = w_1 * 2.5F'_{15}(x) + w_2 * [10F'_9(x) + 100] + w_3 * [2.5F'_1 1(x) + 200]$ $+w_4 * [5e^{-4}F'_{16}(x) + 300] + w_5 * [1e^{-6}F'_1(x) + 400] + 2800$	$\sigma = [10, 20, 30, 40, 50]$
$F_{29} = w_1 * F'_{17}(x) + w_2 * [F'_{18}(x) + 100] + w_3 * [F'_{19}(x) + 200] + 2900$	$\sigma = [10, 30, 50]$
$F_{30} = w_1 * F'_{20}(x) + w_2 * [F'_{21}(x) + 100] + w_3 * [F'_{22}(x) + 200] + 3000$	$\sigma = [10, 30, 50]$
Notes:	
$w_i = \frac{1}{\sqrt{\sum_{j=1}^{D}(x_j - o_{ij})}} exp(-\frac{\sum_{j=1}^{D}(x_i - o_{ij}^2)}{2D\sigma_i^2})$	

5.3.2. University of California, Irvine Dataset (UCI Dataset)

• XOR dataset

The XOR dataset is a classic binary classification problem widely used in machine learning and neural networks [21]. This dataset contains three input features, eight training and testing samples, and one output. The simplicity and nonlinear separability of the XOR dataset make it a standard benchmark for evaluating and validating classification algorithms. It can help researchers and practitioners test and compare the classification capabilities of different algorithms, especially models such as deep learning and neural networks, when dealing with nonlinear problems.

• Balloon dataset

The Balloon dataset is a small dataset for classification problems and has many applications in machine learning and data mining. This dataset contains four features, 16 training and testing samples, and covers two categories [21]. Using the Balloon dataset, researchers can build and test the performance of various classification algorithms, including decision trees, support vector machines, neural networks, and more. The small size of this dataset makes it useful for quickly validating algorithms and debugging classification models. At the same time, the Balloon dataset can also help beginners understand and master fundamental classification problems and algorithms.

• Tic-Tac-Toe dataset

The Tic-Tac-Toe dataset is a commonly used classification dataset for solving the classification problems of the game of tic-tac-toe [21]. It contains nine features and two categories, where the number of training samples is 637 and the number of testing samples is 300. Using the Tic-Tac-Toe dataset, researchers can build and evaluate the performance of various classification algorithms, such as decision trees, logistic regression, support vector machines, etc. This dataset is moderate in size and can be used to quickly verify the accuracy and reliability of the algorithm and provide players with a reference for the next best move position. At the same time, the Tic-Tac-Toe dataset can also be used for teaching purposes to help beginners understand and master the basic concepts and methods of classification problems.

• Heart dataset

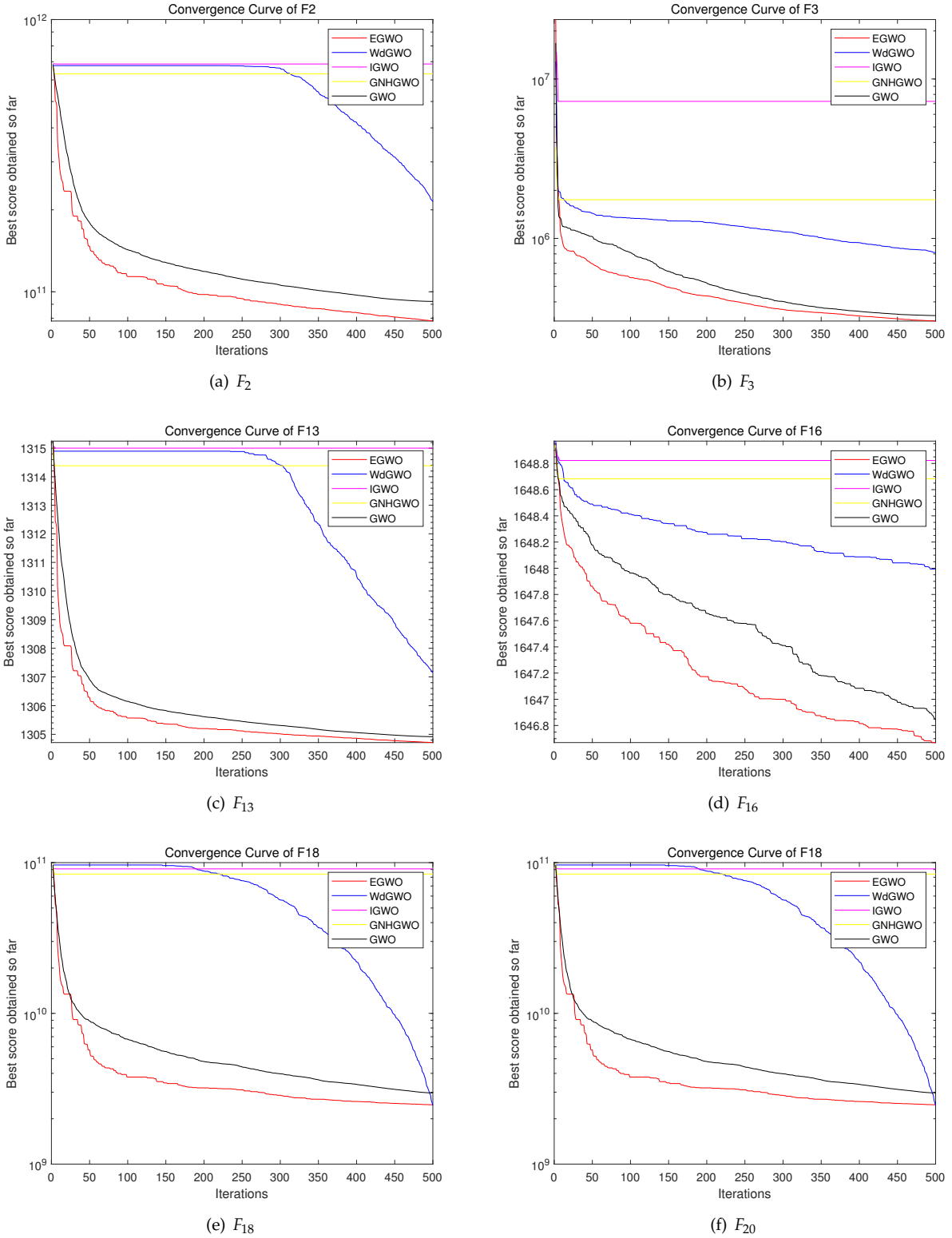
The Heart dataset is a commonly used medical dataset for predicting whether a patient has heart disease [21]. This dataset contains 22 features and two categories, the number of training samples is 80, and the number of test samples is 187. Using the Heart dataset, researchers can better understand and predict the occurrence and risk factors of heart disease and provide clinicians with a basis for auxiliary decision making. In addition, the Heart dataset can also be used for teaching purposes to help students learn and master the basic concepts and techniques of cardiac classification.

6. Analysis and Discussion of Experimental Results

6.1. Analysis and Discussion of Results on IEEE CEC 2014 Benchmark Functions

• Comparison of EGWO with GWO variants

In order to reflect on the advantages of the EGWO optimization algorithm, this part selects the variants of the GWO algorithm, such as GNHGWO, IGWO, and WdGWO. It can be seen from the experimental results in Table 6 that EGWO and GWO have the same average ranking and total ranking and still have certain advantages compared with the other three algorithms. Although EGWO and GWO have the same ranking, it can be seen from Figure 10 that EGWO has a faster convergence speed than the GWO algorithm. Meanwhile, the Friedman ANOVA test and the Wilcoxon rank sum test are selected, and the experimental results shown in Table 7 demonstrate that it is significantly different from the other three variants of GNHGWO, IGWO, and WdGWO.



(a) F_2 (b) F_3

(c) F_{13} (d) F_{16}

(e) F_{18} (f) F_{20}

Figure 10. Convergence curve of EGWO and its variant algorithms.

Table 6. Results of the EGWO and GWO variants.

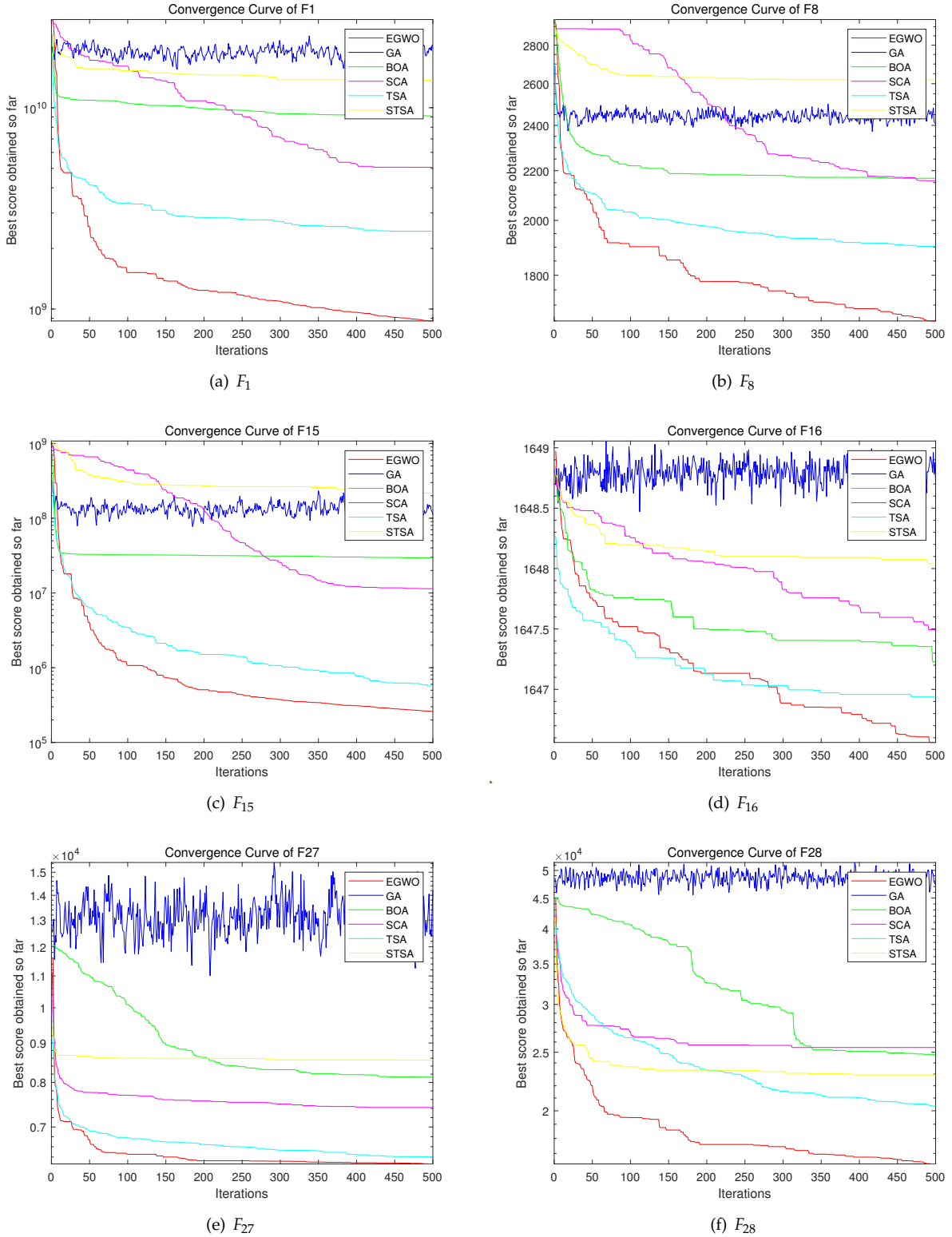
EGWO	GWO	GNHGWO	IGWO	WdGWO	EGWO	GWO	GNHGWO	IGWO	WdGWO
		Mean					**Rank**		
8.851×10^{8}	8.567×10^{8}	2.317×10^{10}	2.441×10^{10}	4.103×10^{9}	2	1	5	3	4
7.809×10^{10}	9.219×10^{10}	6.318×10^{11}	6.867×10^{11}	2.144×10^{11}	1	2	5	3	4
3.033×10^{5}	3.283×10^{5}	1.748×10^{6}	7.235×10^{6}	8.165×10^{8}	1	2	5	3	4
9.873×10^{3}	1.114×10^{4}	2.906×10^{5}	3.237×10^{5}	3.796×10^{4}	1	2	5	3	4
5.214×10^{2}	5.214×10^{2}	5.215×10^{2}	5.215×10^{2}	5.214×10^{2}	5	1	2	3	4
7.269×10^{2}	7.040×10^{2}	7.778×10^{2}	7.793×10^{2}	7.589×10^{2}	2	1	5	3	4
1.482×10^{3}	1.625×10^{3}	6.385×10^{3}	6.697×10^{3}	2.467×10^{3}	1	2	5	3	4
1.728×10^{3}	1.560×10^{3}	2.858×10^{3}	2.917×10^{3}	2.142×10^{3}	2	1	5	3	4
1.867×10^{3}	1.697×10^{3}	3.398×10^{3}	3.489×10^{3}	2.501×10^{3}	2	1	5	3	4
2.176×10^{4}	2.010×10^{4}	3.641×10^{4}	3.647×10^{4}	3.207×10^{4}	2	1	5	3	4
2.433×10^{4}	2.475×10^{4}	3.663×10^{4}	3.669×10^{4}	3.398×10^{4}	1	2	5	3	4
1.205×10^{3}	1.205×10^{3}	1.207×10^{3}	1.207×10^{3}	1.205×10^{3}	1	2	5	3	4
1.305×10^{3}	1.305×10^{3}	1.314×10^{3}	1.315×10^{3}	1.307×10^{3}	1	2	5	3	4
1.603×10^{3}	1.650×10^{3}	3.114×10^{3}	3.259×10^{3}	1.936×10^{3}	1	2	5	3	4
2.360×10^{5}	2.605×10^{5}	6.914×10^{8}	9.421×10^{8}	3.300×10^{7}	1	2	5	3	4
1.647×10^{3}	1.647×10^{3}	1.649×10^{3}	1.649×10^{3}	1.648×10^{3}	1	2	5	3	4
1.168×10^{8}	1.056×10^{8}	3.910×10^{9}	4.650×10^{9}	3.916×10^{8}	2	1	5	3	4
2.473×10^{9}	2.968×10^{9}	8.400×10^{10}	9.090×10^{10}	2.449×10^{9}	5	1	2	3	4
2.691×10^{3}	2.627×10^{3}	2.520×10^{4}	2.911×10^{4}	2.599×10^{3}	5	2	1	3	4
2.477×10^{5}	2.972×10^{5}	5.578×10^{7}	1.200×10^{8}	1.777×10^{6}	1	2	5	3	4
4.545×10^{7}	4.609×10^{7}	1.658×10^{9}	2.561×10^{9}	1.442×10^{8}	1	2	5	3	4
5.753×10^{3}	5.904×10^{3}	2.631×10^{6}	3.533×10^{6}	8.447×10^{3}	1	2	5	3	4
3.275×10^{3}	3.124×10^{3}	1.153×10^{4}	1.349×10^{4}	3.586×10^{3}	2	1	5	3	4
2.927×10^{3}	2.601×10^{3}	4.412×10^{3}	4.524×10^{3}	3.547×10^{3}	2	1	5	3	4
2.898×10^{3}	2.751×10^{3}	4.319×10^{3}	4.495×10^{3}	3.239×10^{3}	2	1	5	3	4
2.816×10^{3}	2.811×10^{3}	4.263×10^{3}	4.593×10^{3}	2.843×10^{3}	2	1	5	3	4
6.265×10^{3}	5.845×10^{3}	1.025×10^{4}	1.144×10^{4}	6.932×10^{3}	2	1	5	3	4
1.633×10^{4}	4.930×10^{3}	4.357×10^{4}	4.455×10^{4}	1.121×10^{4}	2	5	1	3	4
2.203×10^{8}	3.142×10^{3}	9.100×10^{9}	9.814×10^{9}	1.954×10^{8}	2	5	1	3	4
9.028×10^{6}	6.053×10^{3}	5.515×10^{8}	7.087×10^{8}	4.797×10^{6}	2	5	1	3	4
		Average Ranking			1.87	1.87	4.27	3	4
		Total Ranking			1	1	4	2	3

Table 7. Statistical analysis results on EGWO and GWO variants.

		Friedman ANOVA Test					**Wilcoxon Rank Sum Test**		
		SS	***df***	***MS***	***Chi-sq***	***p***	***p***	$\alpha = 0.05$	$\alpha = 0.1$
EGWO vs.	GWO	4374	29	150.828	56.44	0.0017	0.797098	No	No
	GNHGWO	4404	29	151.862	56.83	0.0015	1.73×10^{-6}	Yes	Yes
	IGWO	4409	29	152.034	56.89	0.0015	1.73×10^{-6}	Yes	Yes
	WdGWO	4472	29	154.207	57.7	0.0012	0.00532	Yes	Yes

• Comparison of the GWO with traditional algorithms

Regarding the applicability and interpretability of the EGWO algorithm, this part selects more traditional and popular algorithms, such as GA, BOA, SCA, TSA, and JAYA algorithms. It can be seen from the results in Table 8 that EGWO can achieve the first average ranking and overall ranking. In order to ensure the accuracy of the experimental results, the Friedman ANOVA test and the Wilcoxon rank sum test have verified that EGWO can clearly be distinguished from and compared to other algorithms in Table 9. In addition, the convergence curves of EGWO and GA, BOA, SCA, TSA, and JAYA algorithms, as shown in Figure 11, reflect that the EGWO algorithm can quickly converge and avoid local stagnation.



(a) F_1 (b) F_8

(c) F_{15} (d) F_{16}

(e) F_{27} (f) F_{28}

Figure 11. Convergence curve of EGWO and other traditional algorithms.

Table 8. Results of the EGWO and traditional algorithms.

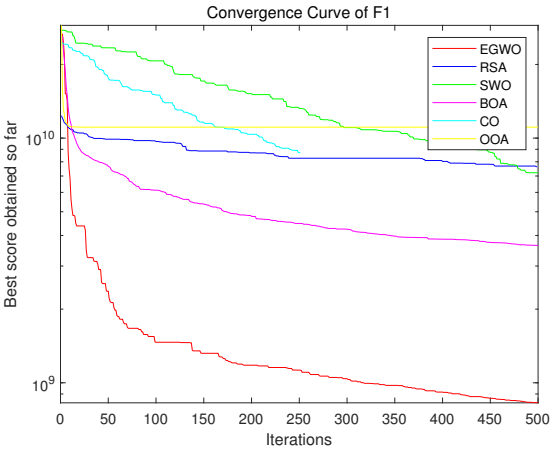
EGWO	GA	BOA	SCA	TSA	JAYA	EGWO	GA	BOA	SCA	TSA	JAYA
Mean						Ranking					
8.851×10^{8}	1.678×10^{10}	9.329×10^{9}	5.084×10^{9}	2.340×10^{9}	1.352×10^{10}	1	5	4	3	6	2
7.809×10^{10}	3.997×10^{11}	3.007×10^{11}	2.301×10^{11}	5.258×10^{10}	4.587×10^{11}	5	1	4	3	2	6
3.033×10^{5}	2.367×10^{7}	3.262×10^{5}	3.991×10^{5}	3.462×10^{5}	9.263×10^{5}	1	3	5	4	6	2
9.873×10^{3}	1.724×10^{5}	1.039×10^{5}	5.127×10^{4}	8.750×10^{3}	1.496×10^{5}	5	1	4	3	6	2
5.214×10^{2}	5.216×10^{2}	5.214×10^{2}	5.214×10^{2}	5.214×10^{2}	5.214×10^{2}	5	4	1	6	3	2
7.269×10^{2}	7.763×10^{2}	7.559×10^{2}	7.607×10^{2}	7.436×10^{2}	7.666×10^{2}	1	5	3	4	6	2
1.482×10^{3}	4.632×10^{3}	3.807×10^{3}	2.992×10^{3}	1.182×10^{3}	4.417×10^{3}	5	1	4	3	6	2
1.728×10^{3}	2.498×10^{3}	2.165×10^{3}	2.171×10^{3}	1.886×10^{3}	2.555×10^{3}	1	5	3	4	2	6
1.867×10^{3}	2.761×10^{3}	2.405×10^{3}	2.415×10^{3}	2.094×10^{3}	3.048×10^{3}	1	5	3	4	2	6
2.176×10^{4}	3.669×10^{3}	3.346×10^{3}	3.196×10^{3}	2.989×10^{3}	3.058×10^{4}	1	5	6	4	3	2
2.433×10^{3}	3.637×10^{3}	3.315×10^{3}	3.348×10^{3}	3.235×10^{3}	3.391×10^{3}	1	5	3	4	6	2
1.205×10^{3}	1.207×10^{3}	1.205×10^{3}	1.205×10^{3}	1.205×10^{3}	1.205×10^{3}	5	1	4	6	3	2
1.305×10^{3}	1.311×10^{3}	1.310×10^{3}	1.308×10^{3}	1.303×10^{3}	1.310×10^{3}	5	1	4	3	6	2
1.603×10^{3}	2.574×10^{3}	2.335×10^{3}	2.041×10^{3}	1.534×10^{3}	2.327×10^{3}	5	1	4	6	3	2
2.360×10^{5}	1.378×10^{8}	2.682×10^{7}	1.075×10^{7}	6.103×10^{5}	2.256×10^{5}	1	5	4	3	2	6
1.647×10^{3}	1.649×10^{3}	1.647×10^{3}	1.648×10^{3}	1.647×10^{3}	1.648×10^{3}	1	5	3	4	6	2
1.168×10^{8}	3.530×10^{9}	1.830×10^{9}	6.445×10^{8}	2.170×10^{8}	1.150×10^{9}	1	5	4	6	3	2
2.473×10^{9}	6.333×10^{10}	4.128×10^{10}	1.405×10^{10}	3.24×10^{3}	1.916×10^{10}	5	1	4	6	3	2
2.691×10^{3}	1.966×10^{4}	1.238×10^{4}	4.594×10^{3}	2.154×10^{3}	8.061×10^{3}	5	1	4	6	3	2
2.477×10^{5}	7.514×10^{7}	1.450×10^{6}	8.578×10^{5}	2.741×10^{5}	6.518×10^{6}	1	5	4	3	6	2
4.545×10^{7}	1.657×10^{9}	5.645×10^{8}	2.578×10^{8}	9.170×10^{7}	4.268×10^{8}	1	5	4	6	3	2
5.753×10^{3}	1.790×10^{6}	4.034×10^{5}	9.476×10^{3}	7.145×10^{3}	3.350×10^{4}	1	5	4	6	3	2
3.275×10^{3}	6.788×10^{3}	2.500×10^{3}	4.224×10^{3}	2.758×10^{3}	5.710×10^{3}	3	5	1	4	6	2
2.927×10^{3}	3.816×10^{3}	2.600×10^{3}	3.166×10^{3}	2.994×10^{3}	3.997×10^{3}	3	1	5	4	2	6
2.898×10^{3}	3.288×10^{3}	2.700×10^{3}	3.093×10^{3}	3.052×10^{3}	3.566×10^{3}	3	1	5	4	2	6
2.816×10^{3}	3.161×10^{3}	2.800×10^{3}	3.053×10^{3}	2.993×10^{3}	3.076×10^{3}	3	1	5	4	6	2
6.265×10^{3}	1.333×10^{4}	8.104×10^{3}	7.421×10^{3}	6.397×10^{3}	8.443×10^{3}	1	5	4	3	6	2
1.633×10^{4}	4.926×10^{4}	2.393×10^{4}	2.564×10^{4}	2.103×10^{4}	2.277×10^{4}	1	5	6	3	4	2
2.203×10^{8}	1.092×10^{10}	3.100×10^{3}	1.837×10^{9}	2.631×10^{7}	1.966×10^{9}	3	5	1	4	6	2
9.028×10^{6}	8.873×10^{8}	3.200×10^{3}	6.262×10^{7}	2.547×10^{6}	1.517×10^{8}	3	5	1	4	6	2
Average Ranking						2.60	3.43	3.70	4.23	4.23	2.80
Total Ranking						1	3	4	5	5	2

Table 9. Statistical analysis results of EGWO and traditional algorithms.

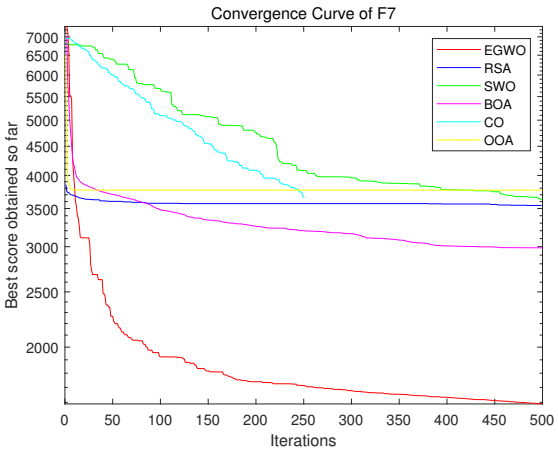
		Friedman ANOVA Test					Wilcoxon Rank Sum Test		
		SS	*df*	*MS*	*Chi-sq*	*p*	*p*	$\alpha = 0.05$	$\alpha = 0.1$
EGWO vs.	GA	4416	29	152.276	56.98	0.0014	1.73×10^{-6}	Yes	Yes
	BOA	4197	29	144.724	54.15	0.0031	0.002279	Yes	Yes
	SCA	4463	29	153.897	57.59	0.0012	3.79×10^{-6}	Yes	Yes
	TSA	4373	29	150.793	56.43	0.0017	0.336552	No	No
	JAYA	4432	29	152.828	57.19	0.0014	3.79×10^{6}	Yes	Yes

• Comparison of GWO with recent algorithms

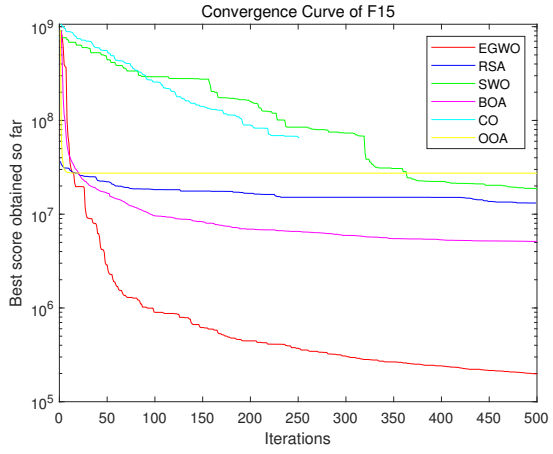
In order to ensure the novelty and performance superiority of the proposed EGWO algorithm, proposed algorithms from the past three years were selected for comparison, such as ZOA, RSA, SWO, BOA, CO, and OOA algorithms. It can be seen from Table 10 that the EGWO algorithm can achieve the best experimental results and ranks first. The Friedman ANOVA test and Wilcoxon rank sum test shown in Table 11 also prove that the EGWO algorithm is superior to the six recently proposed algorithms. Figure 12 shows that EGWO can obtain the fastest convergence speed and smoothly approach the global optimal solution.



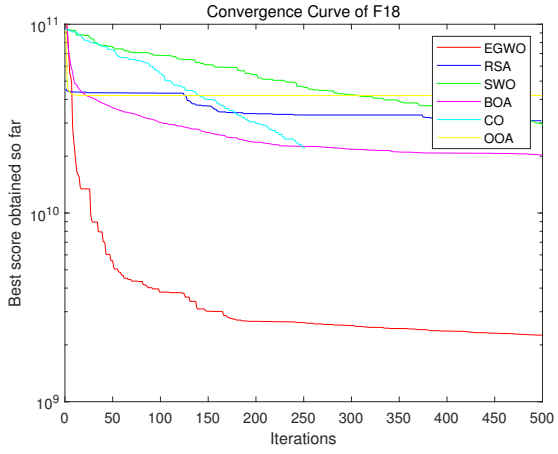
(a) F_1



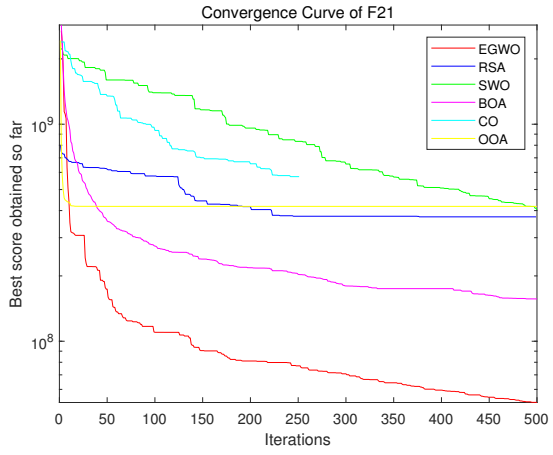
(b) F_7



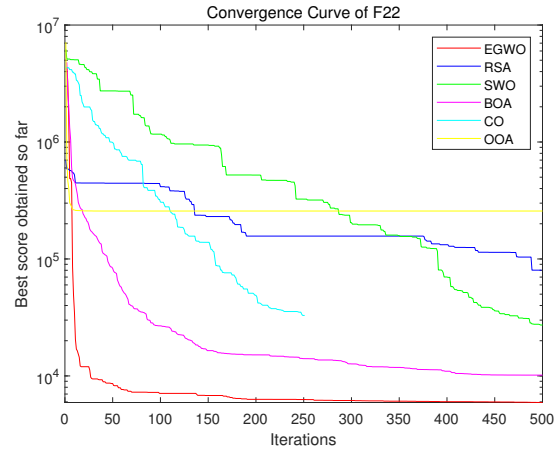
(c) F_{15}



(d) F_{18}



(e) F_{21}



(f) F_{22}

Figure 12. Convergence curve of EGWO and other recent algorithms.

Table 10. Results of the EGWO and recent algorithms.

EGWO	ZOA	RSA	SWO	BOA	CO	OOA	EGWO	ZOA	RSA	SWO	BOA	CO	OOA
Mean							**Rank**						
8.851×10^{8}	9.924×10^{8}	7.443×10^{9}	6.266×10^{9}	4.213×10^{9}	8.570×10^{9}	1.055×10^{10}	1	2	5	4	3	6	7
7.809×10^{10}	1.507×10^{11}	2.803×10^{11}	2.912×10^{11}	2.288×10^{11}	3.154×10^{11}	3.064×10^{11}	1	2	5	3	4	7	6
3.033×10^{5}	2.516×10^{5}	3.070×10^{5}	4.625×10^{5}	3.044×10^{5}	7.333×10^{5}	3.361×10^{5}	2	1	5	3	7	4	6
9.873×10^{3}	2.142×10^{4}	8.052×10^{4}	8.772×10^{4}	5.441×10^{4}	8.940×10^{4}	1.061×10^{5}	1	2	5	3	4	6	7
5.214×10^{2}	5.212×10^{2}	5.214×10^{2}	5.215×10^{2}	5.213×10^{2}	5.215×10^{2}	5.214×10^{2}	2	5	1	3	7	4	6
7.269×10^{2}	7.410×10^{2}	7.574×10^{2}	7.623×10^{2}	7.441×10^{2}	7.657×10^{2}	7.614×10^{2}	1	2	5	3	7	4	6
1.482×10^{3}	2.212×10^{3}	3.554×10^{3}	3.591×10^{3}	3.108×10^{3}	3.671×10^{3}	3.815×10^{3}	1	2	5	3	4	6	7
1.728×10^{3}	1.653×10^{3}	2.254×10^{3}	2.259×10^{3}	2.046×10^{3}	2.308×10^{3}	2.160×10^{3}	2	1	5	7	3	4	6
1.867×10^{3}	1.854×10^{3}	2.372×10^{3}	2.478×10^{3}	2.259×10^{3}	2.725×10^{3}	2.342×10^{3}	2	1	5	7	3	4	6
2.176×10^{4}	2.127×10^{4}	3.095×10^{4}	3.247×10^{4}	2.894×10^{4}	3.137×10^{4}	3.068×10^{4}	2	1	5	7	3	6	4
2.433×10^{4}	2.289×10^{4}	3.154×10^{4}	3.439×10^{4}	2.874×10^{4}	3.461×10^{4}	3.226×10^{4}	2	1	5	3	7	4	6
1.205×10^{3}	1.203×10^{3}	1.205×10^{3}	1.206×10^{3}	1.205×10^{3}	1.206×10^{3}	1.204×10^{3}	2	7	1	3	5	4	6
1.305×10^{3}	1.306×10^{3}	1.309×10^{3}	1.309×10^{3}	1.308×10^{3}	1.309×10^{3}	1.310×10^{3}	1	2	5	3	4	6	7
1.603×10^{3}	1.833×10^{3}	2.236×10^{3}	2.257×10^{3}	2.053×10^{3}	2.285×10^{3}	2.325×10^{3}	1	2	5	3	4	6	7
2.360×10^{3}	1.138×10^{3}	1.439×10^{3}	2.538×10^{3}	7.816×10^{3}	8.170×10^{3}	2.492×10^{3}	1	2	5	3	7	4	6
1.647×10^{3}	1.645×10^{3}	1.647×10^{3}	1.648×10^{3}	1.647×10^{3}	1.648×10^{3}	1.647×10^{3}	2	1	3	5	7	4	6
1.168×10^{8}	1.742×10^{8}	1.208×10^{9}	1.069×10^{9}	5.084×10^{8}	1.156×10^{8}	1.970×10^{8}	1	2	5	4	6	3	7
2.473×10^{9}	7.379×10^{10}	3.391×10^{10}	3.017×10^{10}	2.147×10^{10}	2.261×10^{10}	4.214×10^{10}	1	2	5	6	4	3	7
2.691×10^{3}	3.117×10^{3}	8.969×10^{3}	7.495×10^{3}	5.115×10^{3}	6.173×10^{3}	1.135×10^{4}	1	2	5	6	4	3	7
2.477×10^{5}	2.264×10^{5}	8.816×10^{5}	2.414×10^{6}	6.972×10^{5}	9.080×10^{6}	1.492×10^{6}	2	1	5	3	7	4	6
4.545×10^{7}	6.209×10^{7}	3.956×10^{8}	3.389×10^{8}	1.526×10^{8}	5.455×10^{8}	3.935×10^{8}	1	2	5	4	7	3	6
5.753×10^{3}	7.559×10^{3}	1.104×10^{5}	5.840×10^{4}	1.469×10^{4}	3.293×10^{4}	1.850×10^{5}	1	2	5	6	4	3	7
3.275×10^{3}	2.500×10^{3}	2.500×10^{3}	3.166×10^{3}	2.500×10^{3}	5.278×10^{3}	2.500×10^{3}	2	3	5	7	4	1	6
2.927×10^{3}	2.600×10^{3}	2.600×10^{3}	2.744×10^{3}	2.600×10^{3}	3.579×10^{3}	2.600×10^{3}	2	3	5	7	4	1	6
2.898×10^{3}	2.700×10^{3}	2.700×10^{3}	2.751×10^{3}	2.700×10^{3}	3.501×10^{3}	2.700×10^{3}	2	3	5	7	4	1	6
2.816×10^{3}	2.800×10^{3}	2.800×10^{3}	2.810×10^{3}	2.797×10^{3}	3.407×10^{3}	2.800×10^{3}	5	2	3	7	4	1	6
6.265×10^{3}	7.654×10^{3}	7.556×10^{3}	7.935×10^{3}	3.193×10^{3}	7.507×10^{3}	2.900×10^{3}	7	5	1	6	3	2	4
1.633×10^{4}	2.883×10^{4}	2.382×10^{4}	3.364×10^{4}	4.169×10^{3}	3.102×10^{4}	3.000×10^{3}	7	5	1	3	2	6	4
2.203×10^{8}	2.710×10^{8}	1.935×10^{7}	1.631×10^{9}	2.142×10^{8}	2.929×10^{9}	3.100×10^{3}	7	3	5	1	2	4	6
9.028×10^{6}	2.292×10^{7}	4.422×10^{7}	1.353×10^{8}	3.200×10^{3}	9.792×10^{7}	3.200×10^{3}	5	7	1	2	3	6	4
Averagr Ranking							2.27	2.53	4.20	4.40	4.57	4	6.03
Total Ranking							1	2	4	5	6	3	7

Table 11. Statistical analysis results on EGWO and recent algorithms.

		Friedman ANOVA Test					**Wilcoxon Rank Sum Test**		
		SS	***df***	***MS***	***Chi-sq***	***p***	***p***	$\alpha = 0.05$	$\alpha = 0.1$
EGWO vs.	ZOA	4454	29	153.586	57.47	0.0013	0.047156	Yes	Yes
	RSA	4397	29	151.621	56.74	0.0015	0.000413	Yes	Yes
	SWO	4416	29	152.276	56.98	0.0014	2.84×10^{-5}	Yes	Yes
	BOA	4351	29	150.034	56.14	0.0018	0.022778	Yes	Yes
	CO	4435	29	152.931	57.23	0.0013	1.73×10^{-6}	Yes	Yes
	OOA	4208	29	145.103	54.3	0.003	0.015788	Yes	Yes

Through the above three types of experiments, compared with GWO variants and traditional and popular algorithms proposed in recent years, the experimental results show that the algorithm can achieve the best global optimal solution, and statistical tests also verify the effectiveness of the algorithm's performance and efficiency. The introduction of the mechanism promotes the algorithm to prevent the loss of the optimal solution when the particles migrate and avoid the local optimum simultaneously. The perturbation mechanism ensures the diversity of searches during the three wolves' migration process and improves the exploration performance. The candidate solution mechanism can ensure the ability of exploitation to achieve a balance between exploration and exploitation. The mutual assistance of various mechanisms can strengthen the ability of the EGWO algorithm to find the global optimal solution when solving single-mode, multi-mode, and complex function problems and ensure the exploration and exploitation capabilities of the EGWO algorithm.

6.2. Analysis and Discussion of Results on UCI Dataset

The Tic-Tac-Toe dataset includes nine input features and two categories. According to the results in Table 12, the EGWO-MLP algorithm achieved the highest classification accuracy on this dataset. For the best mean square error (*MSE*) and standard deviation (*Std.*), EGWO-MLP ranks first. The Heart dataset includes 22 features and two categories, and GWO-MLP performs the best, followed by EGWO-MLP, but it still outperforms other comparison algorithms. The XOR dataset contains three input features and one output. By observing the MSE and Rate values in Table 12, it can be seen that the EGWO algorithm achieves the highest classification accuracy and the smallest MSE value on the XOR dataset and Balloon dataset, which contains four input features and two categories. According to the results in Table 12, in terms of the classification rate, the EGWO algorithm has a similar performance to GWO and DE when compared with other algorithms but it is higher than other algorithms. At the same time, EGWO also has a specific stability in terms of the average mean square error (*MSE*) and Std. The training of three datasets shows that the EGWO algorithm has certain advantages in training multi-layer perceptrons and can more stably find the global optimal solution. The introduction of the mechanism prompts EGWO to avoid local optimum and enhances the exploration ability while ensuring the exploitation ability.

Table 12. Training results of EGWO-MLP and other algorithms on the UCI dataset.

	Tic-Tac-Toe Dataset							
	EGWO-MLP	GWO-MLP	DE-MLP	TSA-MLP	PSO-MLP	BA-MLP	GA-MLP	SCA-MLP
Rate	97.643%	93.790%	94.091%	97.310%	87.830%	94.704%	64.725%	93.531%
MSE	0.005	0.001	0.013	0.013	0.017	0.017	0.028	0.015
Std.	2.551	8.245	8.792	5.404	16.561	9.703	33.350	11.274
	Heart Dataset							
	EGWO-MLP	GWO-MLP	DE-MLP	TSA-MLP	PSO-MLP	BA-MLP	GA-MLP	SCA-MLP
Rate	85.292%	89.042%	79.167%	71.417%	59.833%	57.000%	44.750%	70.042%
MSE	0.103	0.076	0.157	0.180	0.272	0.286	0.323	0.206
Std.	33.188	3.074	3.586	3.796	9.042	9.311	8.423	3.543
	XOR Dataset							
	EGWO-MLP	GWO-MLP	DE-MLP	TSA-MLP	PSO-MLP	BA-MLP	GA-MLP	SCA-MLP
Rate	95.417%	93.750%	52.500%	35.417%	31.667%	88.333%	31.667%	47.500%
MSE	0.005	0.010	0.045	0.065	0.191	0.011	0.191	0.090
Std.	8.980	12.607	18.971	17.084	16.973	24.330	16.973	15.186
	Balloon Dataset							
	EGWO-MLP	GWO-MLP	DE-MLP	TSA-MLP	PSO-MLP	BA-MLP	GA-MLP	SCA-MLP
Rate	100%	100 %	100%	44.333%	43.333%	59.000 %	41.167%	97.333%
MSE	3.880×10^{-10}	6.277×10^{-9}	1.07×10^{-6}	0.184	0.197	0.119	0.210	0.001
Std.	0	0	0	12.087	13.792	16.578	14.779	7.397

6.3. Advantages and Disadvantages

Through the two different tests mentioned above, the exploration and exploration capabilities of the EGWO algorithm were verified, and its advantages can be summarized as follows:

- The EGWO algorithm can quickly find the global optimal solution in solving the single-mode simple function problem but can ensure the accuracy of the optimal solution.
- The EGWO-MLP model has apparent advantages in solving multi-classification problems, such as a fast convergence and a strong stability, ensuring a high classification rate.

Although EGWO can have the above advantages, there are still certain shortcomings:

- In combinatorial function problems, local stagnation occurs when searching for the global optimal solution.

- The performance of the EGWO-MLP model in solving single classification problems is not very significant.

7. EGWO-MLP Identification Model

This study investigates the optimization performance of agricultural disease identification using the EGWO algorithm. EGWO trains the MLP to build disease identification models by updating the weights and biases to optimize them continuously. The performance of MLP is improved, and an improvement in the classification rate and a reduction in the error rate are realized.

7.1. Soybean (Large) Dataset

This paper uses the soybean (Large) dataset from the University of California (UCI) Machine Learning Repository dataset, a machine learning database. It is a commonly used standard test dataset. The soybean (Large) dataset has 599 datasets, and this number is still increasing. The soybean (Large) dataset with the 35 attributes shown in Table 13 was used to judge the type and disease of crops and test the model's accuracy. The data source is shown in the Table 13's footer. Furthermore, the data were divided into a test and training set at 70% and 30%, respectively.

Table 13. The attribution of the soybean (Large) dataset.

	Attribution	Means
1	date	April, May, June, July, August, September, October, unknown.
2	plant-stand	normal, lt-normal, unknown.
3	precip	lt-norm, norm, gt-norm, unknown.
4	temp	lt-norm, norm, gt-norm, unknown.
5	hail	yes, no, unknown.
6	crop-hist	diff-lst-year, same-lst-yr, same-lst-two-yrs, same-lst-sev-yrs, unknown.
7	area-damaged	scattered, low-areas, upper-areas, whole-field, unknown.
8	severity	minor, pot-severe, severe, unknown.
9	seed-tmt	none, fungicide, other, unknown.
10	germination	90–100%, 80–89%, lt–80%, unknown.
11	plant-growth	norm, abnorm, unknown.
12	leaves	norm, abnorm.
13	leafspots-halo	absent, yellow-halos, no-yellow-halos, unknown.
14	leafspots-marg	w-s-marg, no-w-s-marg, dna, unknown.
15	leafspot-size	lt-1/8, gt-1/8, dna, unknown.
16	leaf-shread	absent, present, unknown.
17	leaf-malf	absent, present, unknown.
18	leaf-mild	absent, upper-surf, lower-surf, unknown.
19	stem	norm, abnorm, unknown.
20	lodging	yes, no, unknown.
21	stem-cankers	absent, below-soil, above-soil, above-sec-nde, unknown.
22	canker-lesion	dna, brown, dk-brown-blk, tan, unknown.
23	fruiting-bodies	absent, present, unknown.
24	external decay	absent, firm-and-dry, watery, unknown.
25	mycelium	absent, present, unknown.
26	int-discolor	none, brown, black, unknown.
27	sclerotia	absent, present, unknown.
28	fruit-pods	norm, diseased, few-present, dna, unknown.
29	fruit spots	absent, colored, brown-w/blk-specks, distort, dna, unknown.
30	seed	norm, abnorm, unknown.
31	mold-growth	absent, present, unknown.
32	seed-discolor	absent, present, unknown.
33	seed-size	norm, lt-norm, unknown.
34	shriveling	absent, present, unknown.
35	roots	norm, rotted, galls-cysts, unknown.

7.2. Identification Model (EGWO-MLP) Parameter Setting

The EGWO-MLP identification model consists of an input, a hidden, and an output layer. The input layer of EGWO-MLP selects 35 attributes from the UCI soybean (Large) dataset, including the date and germination as input nodes. The hidden layer is set to $(2 \times numberofinputs + 1)$ nodes. In the identification model in this paper, the 35 influenc-

ing factors use soybean as the input of the model, with one hidden layer with a hidden node. Finally, 19 nodes are used as the output of the model.

7.3. Experimental Analysis

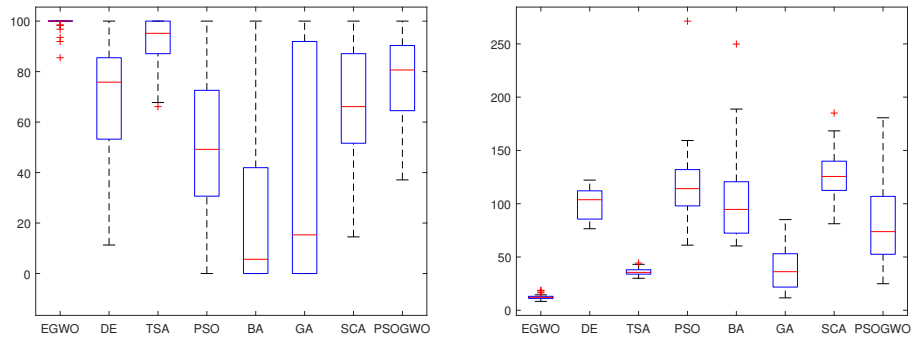
To verify the accuracy of the proposed EGWO-MLP model in identifying soybean disease, some models were accumulated for comparison, such as PSOGWO-MLP, DE-MLP, TSA-MLP, PSO-MLP, BA-MLP, GA-MLP, and SCA-MLP. The final classification and error rates are the criteria to evaluate the performance of the EGWO-MLP model.

It can be seen in Table 14 that the final classification rate of EGWO-MLP is the highest. The EGWO-MLP model can identify diseases more accurately than other methods. This demonstrates that the EGWO algorithm has a strong exploration ability and the ability to reduce local optima. Regarding $\overline{MSE}$, the value obtained by EGWO-MLP ranks first. It can be proven that the EGWO algorithm has a robust searchability. The *Std.* value of EGWO is the smallest, so it can be obtained that the EGWO-MLP model is the most stable and can be effectively applied to the problem of disease identification. In conclusion, the EGWO algorithm can enhance the classification rate and performance of the soybean disease model. The training process of multilayer perceptron leads to EGWO having a strong exploration ability, avoiding local stagnation, and effectively updating the weights and biases of MLP to improve the classification rate.

Table 14. Results of soybean disease identification compared with other algorithms

	EGWO-MLP	PSOGWO-MLP	DE-MLP	TSA-MLP	PSO-MLP	BA-MLP	GA-MLP	SCA-MLP
Rate	98.763%	77.204%	68.548%	91.505%	51.935%	22.957%	39.677%	64.677%
Std.	3.108	15.818	21.933	10.020	27.993	30.217	42.994	23.717
MSE	12.627	80.225	100.142	36.019	118.036	104.711	40.241	125.489
Std.	2.397	38.901	13.384	3.173	38.263	41.688	20.044	22.566

The EGWO-MLP model has certain advantages in disease identification and is superior to PSOGWO-MLP, DE-MLP,TSA-MLP, PSO-MLP, BA-MLP,GA-MLP, and SCA-MLP, as shown in Figure 13. The EGWO-MLP model can obtain a high classification rate and strong robustness. The error rate proves that EGWO has a robust global search ability, effectively balances exploration and exploitation, avoids local stagnation, and improves the convergence speed.



(a) Soybean identification rate (b) Soybean identification MSE

Figure 13. Box diagram of comparative experimental results.

8. Conclusions

It is difficult for existing models to predict actual crop diseases accurately. Therefore, this paper proposes a disease recognition model based on algorithms and MLP. GWO is a swarm intelligence optimization algorithm widely used in many fields, but some shortcomings remain. We proposed the EGWO algorithm based on chaotic disturbance,

candidate mechanisms, and attacking mechanisms. In addition, EGWO is used to optimize the weight and bias items of MLP to construct the EGWO-MLP model.

In order to test the effectiveness of the proposed EGWO algorithm, the IEEE CEC 2014 benchmark functions were compared with the variants of the GWO algorithm, traditional and popular algorithms, and algorithms proposed in recent years. The experimental results show that the EGWO algorithm has certain advantages and stability. In addition, the proposed EGWO-MLP model was verified with four standard classification datasets (XOR, Balloon, Heart, and Tic-Tac-Toe datasets). The experimental results prove that the performance of EGWO-MLP is better than other algorithms, including GWO TSA, PSO, BA, GA, and SCA. Wilcoxon rank sum tests and Friedman ANOVA tests, two statistical tests, proved that at 95% and 90% confidence levels, the EGWO algorithm and other algorithms have apparent advantages.

Meanwhile, the Soybean dataset was selected to verify the effectiveness of disease identification. Compared with PSOGWO-MLP, DE-MLP, TSA-MLP, PSO-MLP, BA-MLP, GA-MLP, and SCA-MLP, EGWO-MLP has a certain degree of accuracy, and it can effectively manage and control crop diseases when they occur.

However, some limitations affect the prediction accuracy, such as the number and attributes of the UCI data referenced by the existing data. After that, a neural network (CNN) was chosen to predict crop diseases based on the dataset. At the same time, we can also select more effective swarm intelligence technologies, optimize and adjust the neural network structure, adjust the parameters, and improve the recognition accuracy.

Author Contributions: Methodology, Q.T. and J.J.; Software, Q.T. and X.M.; Validation, Q.T.; Writing—review & editing, Q.T. and X.M.; Supervision, C.B., H.C., H.W. and W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ratnadass, A.; Fernandes, P.; Avelino, J.; Habib, R. Plant species diversity for sustainable management of crop pests and diseases in agroecosystems: A review. *Agron. Sustain. Dev.* **2012**, *32*, 142–149. [CrossRef]
2. Donatelli, M.; Magarey, R.D.; Bregaglio, S.; Willocquet, L.; Whish, J.P.; Savary, S. Modelling the impacts of pests and diseases on agricultural systems. *Agric. Syst.* **2017**, *155*, 213–224. [CrossRef]
3. Wrather, J.; Anderson, T.; Arsyad, D.; Tan, Y.; Ploper, L.D.; Porta-Puglia, A.; Ram, H.; Yorinori, J. Soybean disease loss estimates for the top ten soybean-producing counries in 1998. *Can. J. Plant Pathol.* **2001**, *23*, 115–121. [CrossRef]
4. Qin, W.; Xue, X.; Zhang, S.; Gu, W.; Wang, B. Droplet deposition and efficiency of fungicides sprayed with small UAV against wheat powdery mildew. *Int. J. Agric. Biol. Eng.* **2018**, *11*, 27–32. [CrossRef]
5. Ficke, A.; Cowger, C.; Bergstrom, G.; Brodal, G. Understanding yield loss and pathogen biology to improve disease management: Septoria nodorum blotch—A case study in wheat. *Plant Dis.* **2018**, *102*, 696–707. [CrossRef] [PubMed]
6. Kulkarni, O. Crop disease detection using deep learning. In Proceedings of the 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA 2018), Pune, India, 16–18 August 2018.
7. Park, H.; JeeSook, E.; Kim, S.-H. Crops disease diagnosing using image-based deep learning mechanism. In Proceedings of the 2018 International Conference on Computing and Network Communications (CoCoNet 2018), Astana, Kazakhstan, 15–17 August 2018.
8. Xiong, Y.; Liang, L.; Wang, L.; She, J.; Wu, M. Identification of cash crop diseases using automatic image segmentation algorithm and deep learning with expanded dataset. *Comput. Electron. Agric.* **2020**, *177*, 105712. [CrossRef]
9. Devi, N.; Sarma, K.K.; Laskar, S. Design of an intelligent bean cultivation approach using computer vision, IoT and spatio-temporal deep learning structures. *Ecol. Inform.* **2023**, *75*, 102044. [CrossRef]
10. Barbedo, J.G.A. Plant disease identification from individual lesions and spots using deep learning. *Biosyst. Eng.* **2018**, *180*, 96–107. [CrossRef]
11. Chen, J.; Chen, J.; Zhang, D.; Sun, Y.; Nanehkaran, Y.A. Using deep transfer learning for image-based plant disease identification. *Comput. Electron. Agric.* **2020**, *173*, 105393. [CrossRef]
12. Goncalves, J.P.; Pinto, F.A.; Queiroz, D.M.; Villar, F.M.; Barbedo, J.G.; Del Ponte, E.M. Deep learning architectures for semantic segmentation and automatic estimation of severity of foliar symptoms caused by diseases or pests. *Biosyst. Eng.* **2021**, *210*, 129–142. [CrossRef]

13. Keyvanpour, M.R.; Shirzad, M.B. Machine learning techniques for agricultural image recognition. In *Application of Machine Learning in Agriculture*; Elsevier: Amsterdam, The Netherlands, 2022; pp. 283–305.
14. Camero, A.; Toutouh, J.; Alba, E. Random error sampling-based recurrent neural network architecture optimization. *Eng. Appl. Artif. Intell.* **2020**, *96*, 103946. [CrossRef]
15. Maurya, L.S.; Hussain, M.S.; Singh, S. Machine learning classification models for student placement prediction based on skills. *Int. J. Artif. Intell. Soft Comput.* **2022**, *7*, 194–207. [CrossRef]
16. Wang, G.; Sim, K.C. Sequential classification criteria for NNs in automatic speech recognition. In Proceedings of the Twelfth Annual Conference of the International Speech Communication Association, Florence, Italy, 27–31 August 2021.
17. Baser, P.; Saini, J.R.; Kotecha, K. Tomconv: An improved cnn model for diagnosis of diseases in tomato plant leaves. *Procedia Comput. Sci.* **2023**, *218*, 1825–1833. [CrossRef]
18. Hush, D.R.; Horne, B.G. Progress in supervised neural networks. *IEEE Signal Process. Mag.* **1993**, *10*, 8–39. [CrossRef]
19. Monti, F.; Boscaini, D.; Masci, J.; Rodola, E.; Svoboda, J.; Bronstein, M.M. Geometric deep learning on graphs and manifolds using mixture model CNNs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 22–25 July 2017; pp. 5115–5124.
20. Srivastava, A.; Singh, A.; Tiwari, A.K. An efficient hybrid approach for the prediction of epilepsy using CNN with LSTM. *Int. J. Artif. Intell. Soft Comput.* **2022**, *7*, 179–193. [CrossRef]
21. Mirjalili, S. How effective is the grey wolf optimizer in training Multi-Layer Perceptrons. *Appl. Intell.* **2015**, *43*, 150–161. [CrossRef]
22. Zhang, C.; Pan, X.; Li, H.; Gardiner, A.; Sargent, I.; Hare, J.; Atkinson, P.M. A hybrid MLP-CNN classifier for very fine resolution remotely sensed image classification. *ISPRS J. Photogramm. Remote Sens.* **2018**, *140*, 133–144. [CrossRef]
23. Das, H.; Jena, A.K.; Nayak, J.; Naik, B.; Behera, H. A novel PSO based Back Propagation learning-MLP (PSO-BP-MLP) for Classification. In *Computational Intelligence in Data Mining-Volume 2, Proceedings of the International Conference on CIDM, 20–21 December 2014*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 461–471.
24. Singh, K.J.; De, T. MLP-GA based algorithm to detect application layer DDoS attack. *J. Inf. Secur. Appl.* **2017**, *36*, 145–153. [CrossRef]
25. Sheikhan, M.; Mohammadi, N. Neural-based electricity load forecasting using hybrid of GA and ACO for feature selection. *Neural Comput. Appl.* **2012**, *21*, 1961–1970. [CrossRef]
26. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Let a biogeography-based optimizer train your Multi-Layer Perceptron. *Inf. Sci.* **2014**, *269*, 188–209. [CrossRef]
27. Emary, E.; Zawbaa, H.M.; Grosan, C.; Hassenian, A.E. Feature subset selection approach by Gray-Wolf Optimization. In Proceedings of the Afro-European Conference for Industrial Advancement, Ababa, Ethiopia, 17–19 November 2015; pp. 1–13.
28. Meng, X.; Jiang, J.; Wang, H. AGWO:Advanced GWO in multi-layer perception optimization. *Expert Syst. Appl.* **2021**, *173*, 114676. [CrossRef]
29. Mittal, N.; Singh, U.; Sohi, B.S. Modified grey wolf optimizer for global engineering optimization. *Appl. Comput. Intell. Soft Comput.* **2016**, *2016*, 7950348. [CrossRef]
30. Zhu, A.; Xu, C.; Li, Z.; Wu, J.; Liu, Z. Hybridizing grey wolf optimization with differential evolution for global optimization and test scheduling for 3d stacked SoC. *J. Syst. Eng. Electron.* **2015**, *26*, 317–328. [CrossRef]
31. Kamboj, V.K. A novel hybrid PSO–GWO approach for unit commitment problem. *Neural Comput. Appl.* **2016**, *27*, 1643–1655. [CrossRef]
32. Gómez, D.; Rojas, A. An empirical overview of the No Free Lunch Theorem and its effect on real-world machine learning classification. *Neural Comput.* **2016**, *28*, 216–228. [CrossRef]
33. Shadkam, E.; Bijari, M. A novel improved cuckoo optimisation algorithm for engineering optimisation. *Int. J. Artif. Intell. Soft Comput.* **2020**, *7*, 164–177. [CrossRef]
34. Mostafa Bozorgi, S.; Yazdani, S. IWOA: An improved whale optimization algorithm for optimization problems. *J. Comput. Des. Eng.* **2019**, *6*, 243–259.
35. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S.; Faris, H. MTDE: An effective multi-trial vector-based differential evolution algorithm and its applications for engineering design problems. *Appl. Soft Comput.* **2020**, *97*, 106761. [CrossRef]
36. Nadimi-Shahraki, M.H.; Taghian, S.; Zamani, H.; Mirjalili, S.; Elaziz, M.A. MMKE: Multi-trial vector-based monkey king evolution algorithm and its applications for engineering optimization problems. *PLoS ONE* **2023**, *18*, e0280006. [CrossRef] [PubMed]
37. Mullen, R.J.; Monekosso, D.; Barman, S.; Remagnino, P. A review of ant algorithms. *Expert Syst. Appl.* **2009**, *36*, 9608–9617. [CrossRef]
38. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
39. Pourpanah, F.; Wang, R.; Lim, C.P.; Wang, X.Z.; Yazdani, D. A review of artificial fish swarm algorithms: Recent advances and applications. *Artif. Intell. Rev.* **2023**, *56*, 1867–1903. [CrossRef]
40. Nadimi-Shahraki, M.H.; Moeini, E.; Taghian, S.; Mirjalili, S. DMFO-CD: A discrete moth-flame optimization algorithm for community detection. *Algorithms* **2021**, *14*, 314. [CrossRef]
41. Rutenbar, R.A. Simulated annealing algorithms: An overview. *IEEE Circuits Devices Mag.* **2023**, *5*, 1867–1903. [CrossRef]
42. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2023**, *179*, 2232–2248. [CrossRef]

43. Li, L.L.; Chang, Y.B.; Tseng, M.L.; Liu, J.Q.; Lim, M.K. Wind power prediction using a novel model on wavelet decomposition-support vector machines-improved atomic search algorithm. *J. Clean. Prod.* **2020**, *270*, 121817. [CrossRef]
44. Erol, O.K.; Eksin, I. A new optimization method: Big bang–big crunch. *Adv. Eng. Softw.* **2006**, *37*, 106–111. [CrossRef]
45. Shaheen, A.M.; Ginidi, A.R.; El-Sehiemy, R.A.; Ghoneim, S.S. A forensic-based investigation algorithm for parameter extraction of solar cell models. *IEEE Access* **2020**, *9*, 1–20. [CrossRef]
46. Askari, Q.; Younas, I.; Saeed, M. Political Optimizer: A novel socio-inspired meta-heuristic for global optimization. *Knowl.-Based Syst.* **2020**, *159*, 105709. [CrossRef]
47. Askari, Q.; Saeed, M.; Younas, I. Heap-based optimizer inspired by corporate rank hierarchy for global optimization. *Expert Syst. Appl.* **2020**, *161*, 113702. [CrossRef]
48. Jiang, J.; Zhao, Z.; Liu, Y.; Li, W.; Wang, H. Dsgwo: An improved grey wolf optimizer with diversity enhanced strategy based on group-stage competition and balance mechanisms. *Knowl.-Based Syst.* **2022**, *250*, 109100. [CrossRef]
49. Duan, Y.; Yu, X. A collaboration-based hybrid GWO-SCA optimizer for engineering optimization problems. *Expert Syst. Appl.* **2023**, *213*, 119017. [CrossRef]
50. Singh, N.; Singh, S. A novel hybrid GWO-SCA approach for optimization problems. *Eng. Sci. Technol.* **2017**, *20*, 1586–1601. [CrossRef]
51. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S.; Zamani, H.; Bahreininejad, A. GGWO: Gaze cues learning-based grey wolf optimizer and its applications for solving engineering problems. *J. Comput. Sci.* **2022**, *61*, 101636. [CrossRef]
52. Malik, M.R.S.; Mohideen, E.R.; Ali, L. Weighted distance grey wolf optimizer for global optimization problems. In Proceedings of the 2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC 2015), Madurai, India, 10–12 December 2015.
53. Long, W.; Jiao, J.; Liang, X.; Tang, M. An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization. *Eng. Appl. Artif. Intell.* **2018**, *68*, 63–80. [CrossRef]
54. Kannan, K.; Yamini, B.; Fernandez, F.M.H.; Priyadarsini, P.U. A novel method for spectrum sensing in cognitive radio networks using fractional GWO-CS optimization. *Ad Hoc Netw.* **2023**, 103135. [CrossRef]
55. Wang, F.; Zhao, S.; Wang, L.; Zhou, Y.; Huang, T.; Shu, X. Study on FOG scale factor error calibration in start-up stage based on GWO-GRU. *Measurement* **2023**, *206*, 112214. [CrossRef]
56. Lim, S.-J. Hybrid image embedding technique using Steganographic Signcryption and IWT-GWO methods. *Microprocess. Microsyst.* **2022**, *95*, 104688.
57. Ocran, D.; Ikiensikimama, S.S.; Broni-Bediako, E. A compositional function hybridization of PSO and GWO for solving well placement optimization problem. *Pet. Res.* **2022**, *7*, 401–408.
58. Yu, X.; Jiang, N.; Wang, X.; Li, M. A hybrid algorithm based on grey wolf optimizer and differential evolution for UAV path planning. *Expert Syst. Appl.* **2014**, *215*, 119327. [CrossRef]
59. Pan, H.; Chen, S.; Xiong, H. A high-dimensional feature selection method based on modified Gray Wolf optimization. *Appl. Soft Comput.* **2023**, *135*, 110031. [CrossRef]
60. Almomani, O. A feature selection model for network intrusion detection system based on pso, gwo, ffa and ga algorithms. *Symmetry* **2020**, *12*, 1046. [CrossRef]
61. Dhal, P.; Azad, C. A multi-objective feature selection method using Newton's law based PSO with GWO. *Appl. Soft Comput.* **2021**, *107*, 107394. [CrossRef]
62. Tu, Q.; Chen, X.; Liu, X. Multi-strategy ensemble grey wolf optimizer and its application to feature selection. *Appl. Soft Comput.* **2019**, *76*, 16–30. [CrossRef]
63. Al-Wajih, A.R.; Abdulkadir, S.J.; Aziz, N.; Al-Tashi, Q.; Talpur, N. Hybridbinary grey wolf with harris hawks optimizer for feature selection. *IEEE Access* **2021**, *9*, 31662–31677. [CrossRef]
64. Abdollahzadeh, B.; Gharehchopogh, F.S. A multi-objective optimization algorithm for feature selection problems. *Eng. Comput.* **2022**, *38*, 1845–1863. [CrossRef]
65. Nikoo, M.; Malekabadi, R.A.; Hafeez, G. Estimating the mechanical properties of Heat-Treated woods using Optimization algorithms-based ANN. *Measurement* **2023**, *207*, 112354. [CrossRef]
66. Astarita, V.; Haghshenas, S.S.; Guido, G.; Vitale, A. Developing new hybrid grey wolf optimization-based artificial neural network for predicting road crash severity. *Transp. Eng.* **2023**, *12*, 100164. [CrossRef]
67. Tian, Y.; Yu, J.; Zhao, A. Predictive model of energy consumption for office building by using improved GWO-BP. *Energy Rep.* **2020**, *6*, 620–627. [CrossRef]
68. Amirsadri, S.; Mousavirad, S.J.; Ebrahimpour-Komleh, H. A levy flightbased grey wolf optimizer combined with back-propagation algorithm for neural network training. *Neural Comput. Appl.* **2018**, *30*, 3707–3720. [CrossRef]
69. Mosavi, A.; Samadianfard, S.; Darbandi, S.; Nabipour, N.; Qasem, S.N.; Salwana, E.; Band, S.S. Predicting soil electrical conductivity using multilayer perceptron integrated with grey wolf optimizer. *J. Geochem. Explor.* **2021**, *220*, 106639. [CrossRef]
70. Al-Badarneh, I.; Habib, M.; Aljarah, I.; Faris, H. Neuro-evolutionary models for imbalanced classification problems. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 2787–2797. [CrossRef]
71. Pasti, R.; de Castro, L.N. Bio-inspired and gradient-based algorithms to train MLPs: The influence of diversity. *Inf. Sci.* **2009**, *179*, 1441–1453. [CrossRef]

72. Al-Majidi, S.D.; Abbod, M.F.; Al-Raweshidy, H.S. A particle swarm optimisation-trained feedforward neural network for predicting the maximum power point of a photovoltaic array. *Eng. Appl. Artif. Intell.* **2020**, *92*, 103688. [CrossRef]
73. Mirjalili, S.; Hashim, S.Z.M.; Sardroudi, H.M. Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. *Inf. Sci.* **2014**, *218*, 11125–11137. [CrossRef]
74. Heidari, A.A.; Faris, H.; Aljarah, I.; Mirjalili, S. An efficient hybrid multilayer perceptron neural network with grasshopper optimization. *Soft Comput.* **2019**, *23*, 7941–7958. [CrossRef]
75. Azzini, A.; Tettamanzi, A.G. Evolutionary ANNs: A state of the art survey. *Intell. Artif.* **2011**, *25*, 19–35. [CrossRef]
76. Alecsa, C.D.; Pinţa, T.; Boros, I. New optimization algorithms for neural network training using operator splitting techniques. *Neural Netw.* **2020**, *126*, 178–190. [CrossRef]
77. Ridge, B.; Gams, A.; Morimoto, J.; Ude, A. Training of deep neural networks for the generation of dynamic movement primitives. *Neural Netw.* **2020**, *127*, 121–131.
78. Zhang, R.; Wang, Q.; Yang, Q.; Wei, W. Temporal link prediction via adjusted sigmoid function and 2-simplex structure. *Sci. Rep.* **2022**, *12*, 16585. [CrossRef] [PubMed]
79. Basterretxea, K.; Tarela, J.M.; del Campo, I. Approximation of sigmoid function and the derivative for hardware implementation of artificial neurons. *IEE Proc.-Circuits Devices Syst.* **2004**, *151*, 18–24. [CrossRef]
80. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S. An improved grey wolf optimizer for solving engineering problems. *Expert Syst. Appl.* **2021**, *166*, 113917. [CrossRef]
81. Akbari, E.; Rahimnejad, A.; Gadsden, S.A. A greedy non-hierarchical grey wolf optimizer for real-world optimization. *Electron. Lett.* **2021**, *57*, 499–501. [CrossRef]
82. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
83. Yang, X.-S.; He, X. Bat algorithm: Literature review and applications. *Int. J. Bio-Inspired Comput.* **2013**, *5*, 141–149. [CrossRef]
84. Kiran, M.S. TSA: Tree-seed algorithm for continuous optimization. *Expert Syst. Appl.* **2015**, *42*, 6686–6698. [CrossRef]
85. Mirjalili, S. SCA: A Sine Cosine algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133. [CrossRef]
86. Arora, S.; Singh, S. Butterfly optimization algorithm: A novel approach for global optimization. *Soft Comput.* **2019**, *23*, 715–734. [CrossRef]
87. Abdel-Basset, M.; Mohamed, R.; Jameel, M.; Abouhawwash, M. Spider wasp optimizer: A novel meta-heuristic optimization algorithm. *Artif. Intell. Rev.* **2023**, 1–64. [CrossRef]
88. Trojovská, E.; Dehghani, M.; Trojovský, P. Zebra optimization algorithm: A new bio-inspired optimization algorithm for solving optimization algorithm. *IEEE Access* **2022**, *10*, 49445–49473. [CrossRef]
89. Abualigah, L.; Abd Elaziz, M.; Sumari, P.; Geem, Z.W.; Gandomi, A.H. Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer. *Expert Syst. Appl.* **2022**, *191*, 116158. [CrossRef]
90. Prakash, T.; Singh, P.P.; Singh, V.P.; Singh, S.N. A novel Brown-bear optimization algorithm for solving economic dispatch problem. In *Advanced Control & Optimization Paradigms for Energy System Operation and Management*; River Publishers: New York, NY, USA, 2023; pp. 137–164.
91. Dehghani, M.; Trojovský, P. Osprey optimization algorithm: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems. *Front. Mech. Eng.* **2023**, *8*, 1126450. [CrossRef]
92. Akbari, M.A.; Zare, M.; Azizipanah-Abarghooee, R.; Mirjalili, S.; Deriche, M. The cheetah optimizer: A nature-inspired metaheuristic algorithm for large-scale optimization problems. *Sci. Rep.* **2022**, *8*, 10953. [CrossRef] [PubMed]
93. Liang, J.J.; Qu, B.Y.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*; Technical Report; Computational Intelligence Laboratory, Zhengzhou University: Zhengzhou, China; Nanyang Technological University: Singapore, 2013; Volume 635.